2011-05-05

Inertial System Modeling and Kalman Filter Design from Sensor Specifications with Applications in Indoor Localization

Matthew Lowe
Worcester Polytechnic Institute

Inertial System Modeling and Kalman Filter Design from Sensor Specifications with Applications in Indoor Localization

by

Matthew S. Lowe

A Thesis
Submitted to the Faculty
of the
WORCESTER POLYTECHNIC INSTITUTE
in partial fulfillment of the requirements for the
Degree of Master of Science
in
Electrical and Computer Engineering
by

May 5, 2011

APPROVED:

Professer David Cyganski, Major Advisor

Professor R. J. Duckworth

Professor Arthur Heinricher

Abstract

This thesis presents a 6 degree of freedom (DOF) position and orientation tracking solution suitable for pedestrian motion tracking based on 6DOF low cost MEMS inertial measurement units. This thesis was conducted as an extension of the ongoing efforts of the Precision Personnel Location (PPL) project at WPI. Prior to this work most of the PPL research focus has been on Radio Frequency (RF) location estimation. The newly developed inertial based system supports data fusion with the aforementioned RF system in a system currently under development.

This work introduces a methodology for the implementation of a position estimation system based upon a Kalman filter structure, constructed from industry standard inertial sensor specifications and analytic noise models. This methodology is important because it allows for both rapid filter construction derived solely from specified values and flexible system definitions. In the course of the project, three different sensors were accommodated using the automatic design tools that were constructed.

This thesis will present the mathematical basis of the new inertial tracking system followed by the stages of filter design and implementation, and finally the results of several trials with actual inertial data captures, using both public reference data and inertial captures from a foot mounted sensor that was developed as part of this work.

Acknowledgements

My family: They have supported my efforts, academic and otherwise, for more years than I can remember and in more ways than I can count.

My sponsor: Without the support of the U.S. Army Natick Soldier Systems Center and Department of Homeland Security, this simply couldn't have happened. It is a tribute to their wisdom in recognizing the potential of this research.

My fellow team members: To past members who have moved on, their original work has paved the way, and they never cease to be an inspiration. To those who are still here whose help in system tests and hardware development cannot be over appreciated. To Bob Boisse who keeps the ship afloat with his organization and technical skills.

My committee: Professor Duckworth's and Dean Heinricher's incredible dedication, especially towards the end when time was short and their schedules tight, has greatly enriched this work.

My advisor: My thanks to Professor Cyganski, whose depth of patience must have been put to the test, and who has the great talent to give good advice in times of need without stifling creativity.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This thesis develops a framework for the construction of an estimator for the motion of a body observed by a 6 degree of freedom inertial measurement unit (IMU), comprising 3 gyroscopes and 3 accelerometers. The purpose of this work is to support the work being done by the Precision Personal Location (PPL) project at Worcester Polytechnic Institute (WPI). The project and thesis have been supported by grants from the U.S. Army Natick Soldier Systems Center and the Department of Homeland Security. The overarching goal of the project is to create an indoor tracking and navigation system for first responders with the capability to locate their personnel in the case of injury and prevent them from becoming disoriented. This thesis takes the project in a new direction by implementing an inertial tracking element and the means to obtain a Kalman Filter estimator design directly from IMU specifications.

1.1 Current PPL Radio Frequency System and Path Forward

The previous PPL radio frequency (RF) location system, which has been developed, is described in [2]. For a discussion of the current progress on the project see [3, 5]. For an example of how information from different location systems can be merged for improved results see [7, 6].

Results typical of the RF based location system developed in the previous work can be seen in Figure 1.1. The figure is captures the outcomes of a trial of the RF location system, which utilizes σART [2]. In the test depicted, the mobile transmitter was moved in a closed path inside a single family dwelling, crossing three rooms, with internal and

external walls, shown as thick black lines. In the figure the blue squares indicate truth points and the blue line, an approximate truth path. RF measurements were made by a set of antennas surrounding the building and a set of synchronized receivers. Notice that the errors, which may appear to be large at first, do not grow in time. This is because the errors in the RF data set are largely caused by effects in the channel, either multipath or path delays, and simple noise on the received signal, either due to external sources or receiver front end electronics. The location estimates are only dependent on received signals at that instance in time and hence position estimate errors are not cumulative. That is, the errors are largely dependent on the position of the RF mobile unit and not the duration of time that has passed since the beginning of the test, or distance traveled.

A similar test was performed with an inertial unit and the linear accelerations and rotational velocity measurements were simply integrated (i.e. a strict application of Newtonian Physics) obtaining results shown in Fig. 1.2. Notice that although the system starts with a small error it grows rapidly throughout the test. This problem, the accumulation of error, will be partially alleviated with additional information indicating moments of zero velocity which can be obtained by strapping the IMU to a foot and observing the moments of the foot's contact with the ground. These zero velocity updates (ZUPTS) allow a more sophisticated estimator to track and correct for accumulated errors. In spite of this use of ZUPTS, error residuals will still accumulate, eventually leading to path divergence. It is our hope that with the addition of the RF based estimates, whose error does not grow with time or distance traveled, this issue can be resolved.

The goal of the work documented in this thesis is to formulate a design process for integrating navigation systems such that an Inertial Measurement Unit (IMU) based tracking capability can be explored as a complement to our current RF based location tracking to improve tracking performance. The two systems would appear to be very complementary, as the RF systems error does not grow in the long term whereas an IMU based system's performance, though accurate in the short term, declines in the long term due to the accumulation of error. In this thesis we will be discussing primarily the implementation of a Kalman structure estimation/filtering system capable of processing the inertial data provided by the IMU to obtain a navigation solution. Many of the previous works in this field have been focused on designing systems and rely on tweaking them until they achieve acceptable results. In this work we attempt to develop all system parameters from models for the inertial sensor units that employ only published performance data, such as Allan

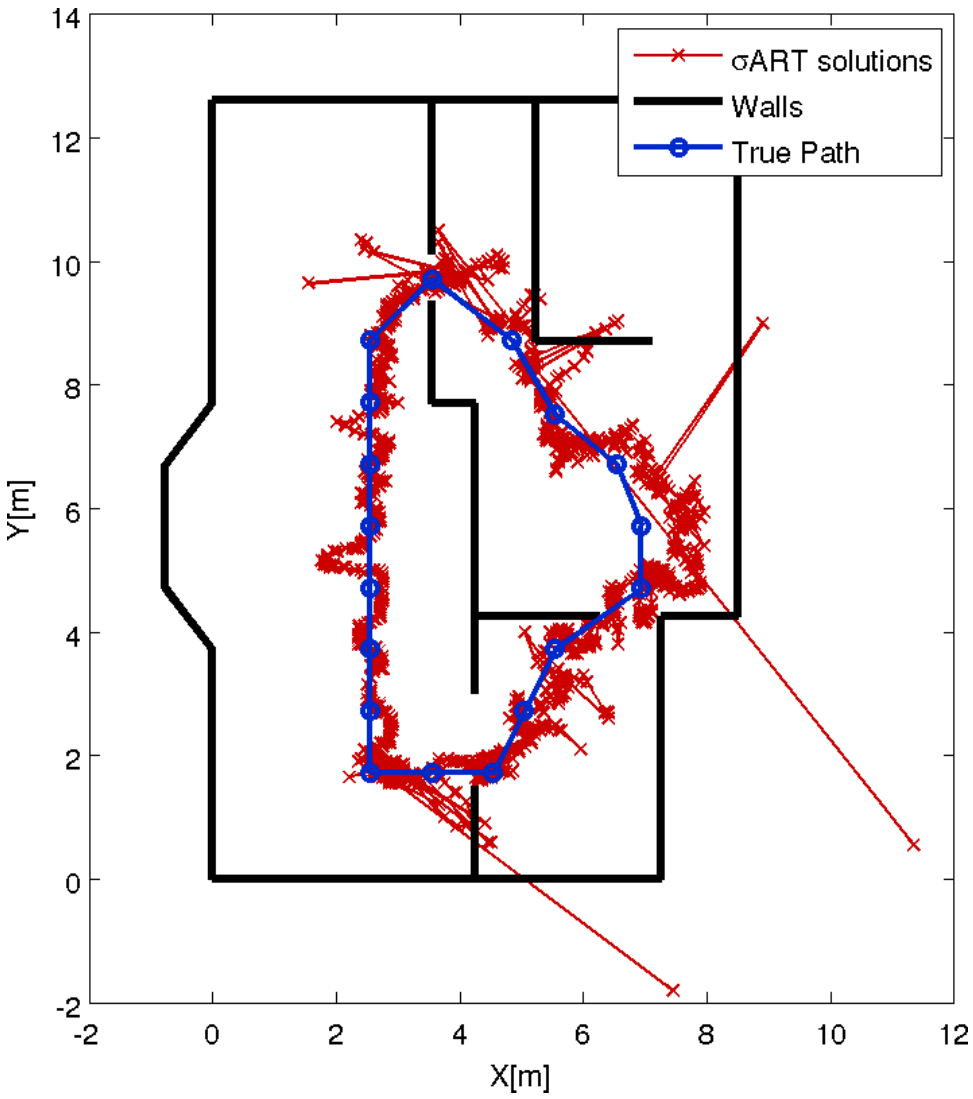

Figure 1.1: Religious Center Radio Frequency System Results

variance curves, and eschew the manual adjustment of parameters.

1.1.1 The Flow of the Thesis

This thesis outlines the steps necessary to realize our concept of tracking an object with the IMU from design to testing with actual data set examples. The IMUs chosen for implementation of the system will be low cost Microelectromechanical systems (MEMS) in a strap-down configuration, attached to the body we wish to track. We will achieve our objective of tracking the body by creating a state space model of the entire system; the

model's framework is detailed in Chapter 2. We begin by defining the system movement in Section 1.2 and how to represent its rotation in Chapter 3. This definition of motion will be transformed into a linear discrete state space model in Chapter 4 through linearization and discretization. Additionally we will create state space models for the IMU's sensors from their specified Allan Deviations in Chapter 5. We will discuss the processing technique we will be using, the Kalman Filter, in Chapter 6. The method of combining our models for motion and IMU measurements will be done in Chapter 7. Finally, results of this method of modeling and processing will be shown in Chapters 8 and 10. A visual representation of this flow can be seen in Figure 1.3

1.1.2 Motivation for the Flow of the Thesis

As previously stated, our goal is to design a system capable of taking an IMU's measurements and providing position estimates, which can be seen in Fig. 1.4.

For our work we will be specifically considering the group of IMUs commonly referred to as six degree of freedom low-cost MEMS, like the ADIS16375 shown in Fig. 1.5.

The unit is said to have six degrees of freedom because it measures the three orthogonal linear accelerations and the three orthogonal rotational velocities, which are shown in Figs. 1.6 and 1.7 respectively.

This unit will be implemented in a strap down configuration, attached to the body we wish to track. In order to be able to cancel many of the errors the system accumulates we will be mounting the unit to the individual's foot so that the unit experiences well defined periods of zero motion, when the foot is planted on the floor. We call these periods Zero Velocity Updates (ZUPTs) and they have been shown to greatly increase tracking accuracy [9]. An example unit in a strap down configuration on the foot can be seen in Fig. 1.8.

The system we will be implementing in order to process the measurements made by the IMU is a Kalman Filter. The Kalman Filter uses an algorithm for finding the Minimum Mean Squared Error estimator of a state space based signal in White Gaussian Noise. The state space model is shown in Fig. 1.9 and discussed at length in Chapter 2.

Once a system a has been described by a discrete, linear, state space model the Kalman Filter can simply be implemented to process data. The majority of this work, therefore, will be in modeling our system as a discrete, linear, state space model. This will take place in two major developments. The first is to describe the motions of the object with

a set of continuous time nonlinear differential equations, which arises almost directly from the definitions of motion. This comprises both the physical position of the object and the rotation of its own, local, coordinate frame. Some focus will be needed on the description of rotation because, as we shall see, it is not readily moved into a state space model. At the end of this process we will have a model in the following form,

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(t, \boldsymbol{x}, \boldsymbol{y})$$

$\boldsymbol{x}$ is state of the system and

$\boldsymbol{y}$ is forcing function on the system.

By making a series of approximations we are going to be able to first linearize and then discretize this nonlinear continuous time model into a linear discrete time model. The final system model form will be that of the Kalman Filter's state space system,

$$\boldsymbol{x}[i+1] = \boldsymbol{\Phi}[i]\boldsymbol{x}[i] + \boldsymbol{\Gamma}[i]\boldsymbol{u}[i] + \boldsymbol{\Lambda}[i]\boldsymbol{w}[i].$$

The make up of this equation can be seen in Fig. 1.10.

The other half of the problem is building a model for the measurements made by sensors such as the IMU. In this regard there are two general issues that we need to deal with. Starting with a generic measurement model of the following form,

$$\zeta[i] = g[i, x[i]] + \mu[i]$$

g is some, potentially nonlinear function, that 'observes' current state and

μ is some measurement error/noise.

We are going to have to reformulate this into the expected state space form of,

$$z[i] = \boldsymbol{H}[i]\boldsymbol{x}[i] + n[i].$$

To achieve this formulation we will need to first linearize the function g and then build a model for the noise μ, noting that the noise may not, and most likely will not, be the White Gaussian Noise required of n. The linearization process is carried out much like it was for the motion describing system, using an estimated point $\tilde{\boldsymbol{x}}$. To formulate the noise model however we will be using a technique based on the observation that most IMU based sensors have similar noise characteristics that we can identify from their Allan deviation/variance plots. From these characteristics we will be able to identify a sequence of linear discrete

time models that will allow our system to track the non-Gaussian portions of the noise, an example characterization can be seen in Fig. 1.11.

The result will be a noise model composed of three basic types; bias walk, β, flicker noise, κ, and WGN, n. The entire observation process is shown in Fig. 1.12.

Our final system can be visualized as comprising all the components shown in Fig. 1.13.

1.2 System Description

The core of this thesis is dedicated to the development and implementation of a strap-down inertial tracking system. That is the Inertial Measurement Unit (IMU) is physically strapped onto the object of interest, a first responder in our case. So by tracking the motion of the IMU we also track the motion of the first responder. Because the unit moves with first responder we must track its current coordinate frame $\mathcal{C}_\ell$ in reference to our own $\mathcal{C}$ so we can properly relate its measurements.

We are interested in tracking the motion of the object, who's path is taken to be $\boldsymbol{p}(t)$, through a coordinate frame $\mathcal{C}$, attached to a house for instance. This body will be observed by accelerometers so its motions are expanded through Newtonian physics to include a differential description of acceleration, the process is listed in (1.1).

$$\dot{\boldsymbol{p}}(t) = \boldsymbol{v}(t) \tag{1.1a}$$

$$\dot{\boldsymbol{v}}(t) = \boldsymbol{a}(t) \tag{1.1b}$$

$$\dot{\boldsymbol{a}}(t) = \boldsymbol{j}(t) \tag{1.1c}$$

In these equations $\boldsymbol{p}(t)$ denotes position, $\boldsymbol{v}(t)$, velocity, $\boldsymbol{a}(t)$, acceleration, and $\boldsymbol{j}(t)$, jerk. Attached and moving with the body is the sensor package in a local coordinate frame, $\mathcal{C}_\ell$, which is characterized by its current position, $\boldsymbol{p}(t)$, and rotation, $\boldsymbol{S}(t)$, in relation to the global coordinate system $\mathcal{C}$. The two coordinate frames can be seen in Fig. 1.14, with the $\hat{x}$, $\hat{y}$, $\hat{z}$ being used to denote the axes with respect to the coordinates $x(t)$, $y(t)$, $z(t)$ are measured. An example IMU sensor can be seen in Fig. 1.15.

Observations of the local coordinate frame, denoted by their ℓ subscript, can be converted to their global coordinate frame equivalents, with no subscript, by the following equations.

$$\boldsymbol{S}(t)(\boldsymbol{v_\ell}(t)) = \boldsymbol{v}(t) \tag{1.2a}$$

$$\boldsymbol{S}(t)(\boldsymbol{a_\ell}(t)) = \boldsymbol{a}(t) \tag{1.2b}$$

$$\boldsymbol{S}(t)(\boldsymbol{j_\ell}(t)) = \boldsymbol{j}(t) \tag{1.2c}$$

$$\tag{1.2d}$$

The IMU in the local coordinate frame makes measurements of the local acceleration and rate of rotational change or angular velocity. From the IMU's measurements we hope to be able to track the body's position through the integration of these measured quantities.

The sensor inherently measures physical effects from the Earth's rotation and gravitational potential gradient along with unknowable errors, referred to as noise. Acceleration due to gravity is expected from the equivalence principal of physics and is also dependent on the local variations in gravity. The effects of the rotation of the Earth and by extension our coordinate frames $\mathcal{C}$ and $\mathcal{C}_\ell$, the Coriolis effect, causes a constant rotation to be measured by the gyroscopes. These effects are described on page 99 of [1]. The effect of gravity will need to be tracked but on the scale of time, velocity, and distance with which we will be dealing, the effects of Earth's rotation can and will be ignored in order to simplify the system. The resulting sensor dynamics of our system will be described in Eq. (1.3).

1.2.1 IMU Measurements

The IMU takes measurements of the current acceleration in the local frame, $\tilde{\boldsymbol{\zeta}}(t)$, and the current rotation rate, $\tilde{\dot{\boldsymbol{S}}}_\ell(t)$.

$$\boldsymbol{\zeta}(t) = \boldsymbol{a_\ell}(t) + \boldsymbol{S}(t)^{-1}\begin{pmatrix}0\\0\\g\end{pmatrix} + \breve{\boldsymbol{\mu}} \tag{1.3a}$$

$$\tilde{\dot{\boldsymbol{S}}}_{\boldsymbol{\ell}}(t) = \dot{\boldsymbol{S}}_{\boldsymbol{\ell}}(t) + \breve{\boldsymbol{\mu}} \tag{1.3b}$$

The noise, $\boldsymbol{\mu}$, is discussed in Chapter 5.

The rotational dynamics are described by

$$\dot{\boldsymbol{S}}(t) = \boldsymbol{S}(t)\Big(\dot{\boldsymbol{S}}_{\boldsymbol{\ell}}(t)\Big).$$

We will instead track the inverse rotation $\boldsymbol{S}(t)^{-1}$, which when written in matrix form is the matrix transpose as we will see in Chapter 3.

$$\dot{\boldsymbol{S}}(t)^{\mathrm{T}} = \dot{\boldsymbol{S}}_{\boldsymbol{\ell}}(t)^{\mathrm{T}}\boldsymbol{S}(t)^{\mathrm{T}} \tag{1.4}$$

The dynamics and conversions between coordinates for the system can be found in Eq. (1.1) and Eq. (1.2) with the additional identity Eq. (1.4).

1.3 Conventions

Throughout the thesis we will be following a static set of conventions. Acronyms will be defined as they come up and additionally a complete list is provided in Table 1.1. Mathematical conventions are listed in Table 1.2 and Table 1.3 with the following notes and exceptions:

1. Vectors will be represented as bold lowercase letters and will be represented by column matrices.

2. Matrices will be represented by bold capital letters.

3. Functions will be defined so as to indicate their range and domain: their name implying their range and their input variables defining their domain, see the table for examples.

4. Fraktur lettering, $\mathfrak{A}$, $\mathfrak{B}$, $\mathfrak{a}$, $\mathfrak{b}$, will be used to emphasize that one or more of the parameters is a function, as without their prior definitions the function f and the real variable f would look identical.

5. After a function $f(t)$ is defined it will be referred to as f when the function itself is implied and $f(t)$ when an evaluation is implied. For example, $g(f(t))$ is the function g evaluated at the evaluation of f at t, whereas $\mathfrak{G}(f)$ is the evaluation of the functional, $\mathfrak{G}$, at the function f. The exception to this rule is in the taking of partial derivatives; if a function is given by $\boldsymbol{F}(r(t)) = r(t)^2$ then in the expression $\frac{\partial \boldsymbol{F}}{\partial r}$ it is clear that the derivative is in terms evaluated function $r(t)$ not the function itself, r, so in our case $\frac{\partial \boldsymbol{F}}{\partial r}(a) = 2a$. The partial may also be shortened to a subscript $\boldsymbol{F}_r(a)$, which is different from the notation for indexing only in the variable used. Variable overlap is avoided to mitigate this potential confusion.

6. A fully indexed vector, matrix, or function may drop its bold typeface in order to emphasize that it is a singleton.

7. In order to reduce the number of variables various 'decorations' are used, their form and description can be found in Table 1.3.

8. In particular the use of the breve accent, $\breve{\boldsymbol{x}}$, needs mention attention. This is used whenever a particular variable has been reduced to a set of indices which are relevant. For example if a vector $\boldsymbol{x}$ contains a set of subindices, $\boldsymbol{a}$, having to do with a particular subsystem in discussions, relating to that subsystem, $\breve{\boldsymbol{x}}$ may be used to indicate only those portions of $\boldsymbol{x}$, $\breve{\boldsymbol{x}} = \boldsymbol{x}_{\boldsymbol{a}}$.

9. Decorations may be stacked with meaning derived from the bottom up. So $\tilde{\dot{\boldsymbol{x}}}$ is an estimate of the derivative of $\boldsymbol{x}$ whereas $\dot{\tilde{\boldsymbol{x}}}$ is the derivative of the estimate.

10. The script variant ℓ of l will be used as a subscript at some points to indicate a local coordinate system version of a variable and is not meant as index into a vector or a matrix.

11. The subscript 0 will be used to describe a unique variable and should not be confused with an index subscript and in many cases an initial value is implied, as in $\boldsymbol{x}_0$. Also t, used always as 'time', possess the special property that whenever it has a subscript it is a new variable and never a vector, so t_1 and t_2 are two different times and never in reference to a vector $\boldsymbol{t}$.

3D	3 Dimensions
IMU	Inertial Measurement Unit
MEMS	Microelectromechanical systems
pdf	Probability Density Function
PPL	Precision Personal Location
RF	Radio Frequency
WGN	White Gaussian Noise
WPI	Worcester Polytechnic Institute
ZUPT	Zero Velocity Update

Table 1.1: Table of Acronyms

We will now begin by describing the general state space model we will be using.

Real Numbers	$\mathbb{R}$
n-dimensional Real Space	$\mathbb{R}^n$
$n \times m$ Matrix Space	$\mathbb{R}^{n \times m}$
Vector	$\boldsymbol{x}$, $\boldsymbol{y}$, $\boldsymbol{\lambda}$
Matrix	$\boldsymbol{A}$, $\boldsymbol{Q}$, $\boldsymbol{\Xi}$
Identity Matrix	$\boldsymbol{I}$
Zero Matrix	$\boldsymbol{0}$
Transpose	$.^{\mathrm{T}}$
Column Indexed	$\boldsymbol{A}_j$, $\boldsymbol{A}_{j,:}$
Row Indexed	$\boldsymbol{A}_{:,k}$, $\boldsymbol{x}_k$, x_k
Column and Row Indexed	$\boldsymbol{A}_{j,k}$, $A_{j,k}$
Function	$f(t)$ implies $f : \mathbb{R} \to \mathbb{R}$
Vector Valued Function	$\boldsymbol{f}(t)$ implies $f : \mathbb{R} \to \mathbb{R}^n$
Matrix Valued Function	$\boldsymbol{\Xi}(t)$ implies $f : \mathbb{R} \to \mathbb{R}^{n \times m}$
Function of Vectors	$f(\boldsymbol{x})$ implies $f : \mathbb{R}^n \to \mathbb{R}$
Expected Value	$E(X)$ is the Expected Value of X
Functionals	$\mathfrak{J}(f)$ implies $\mathfrak{J} : (\mathbb{R} \to \mathbb{R}) \to \mathbb{R}$ where $f : \mathbb{R} \to \mathbb{R}$
Time Derivative	$\frac{df}{dt}(t) = \dot{f}(t)$
Partial Derivative	$\frac{\partial f}{\partial a}(t) = f_a(t)$
Indexed Vector Function	$\boldsymbol{f}(t)_k$
Vector Derivative	$\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}(\boldsymbol{x}) = \begin{pmatrix} \frac{\partial \boldsymbol{f}}{\partial x_1}(\boldsymbol{x})_1 & \frac{\partial \boldsymbol{f}}{\partial x_2}(\boldsymbol{x})_1 & \cdots \\ \frac{\partial \boldsymbol{f}}{\partial x_1}(\boldsymbol{x})_2 & \frac{\partial \boldsymbol{f}}{\partial x_2}(\boldsymbol{x})_2 & \cdots \\ \vdots & & \end{pmatrix}$
Discrete Functions	$f[n]$, $\boldsymbol{f}[n]$, $\boldsymbol{F}[n]$

Table 1.2: Table of Math Conventions

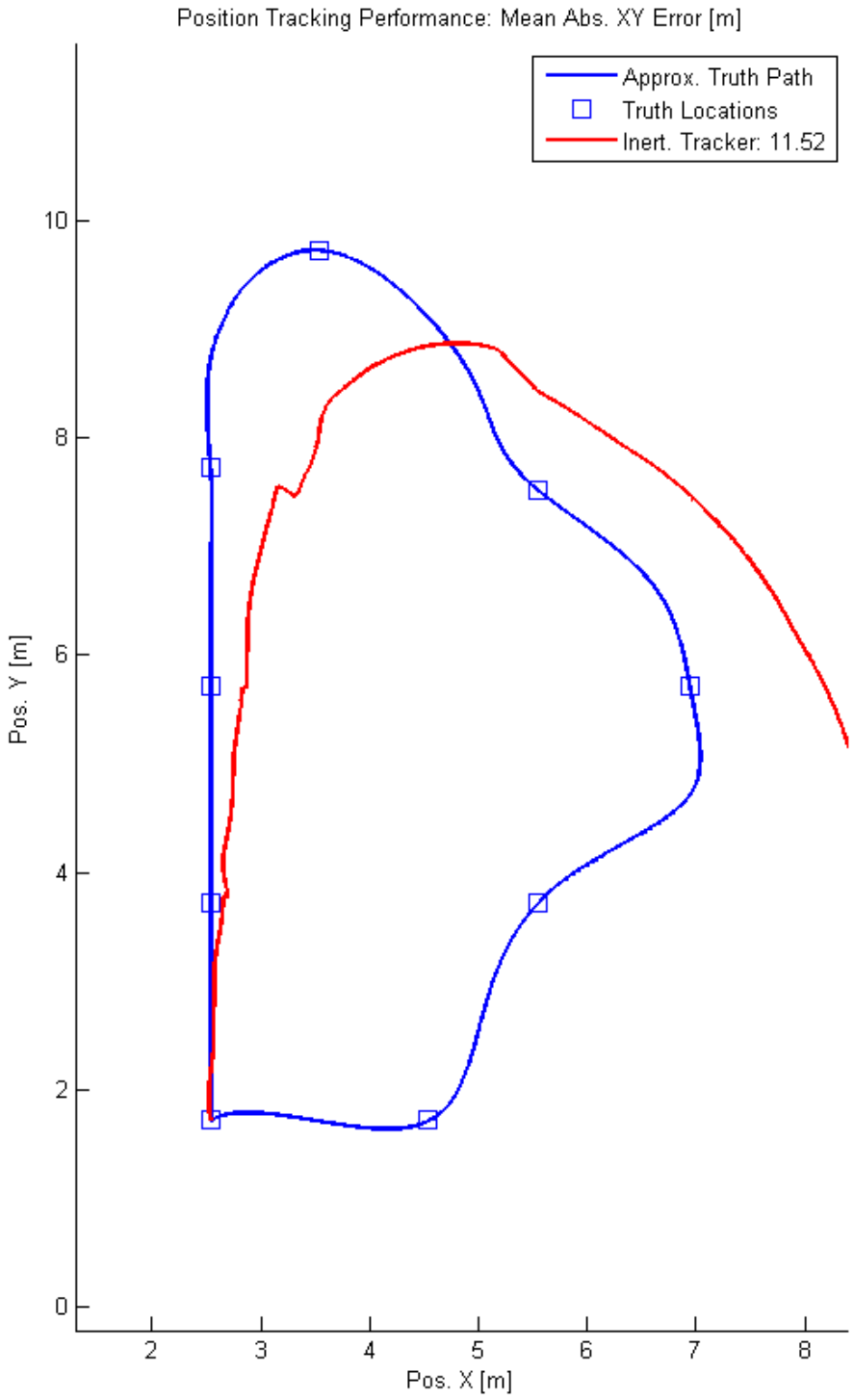

Figure 1.2: Religious Center Simple Inertial System Results

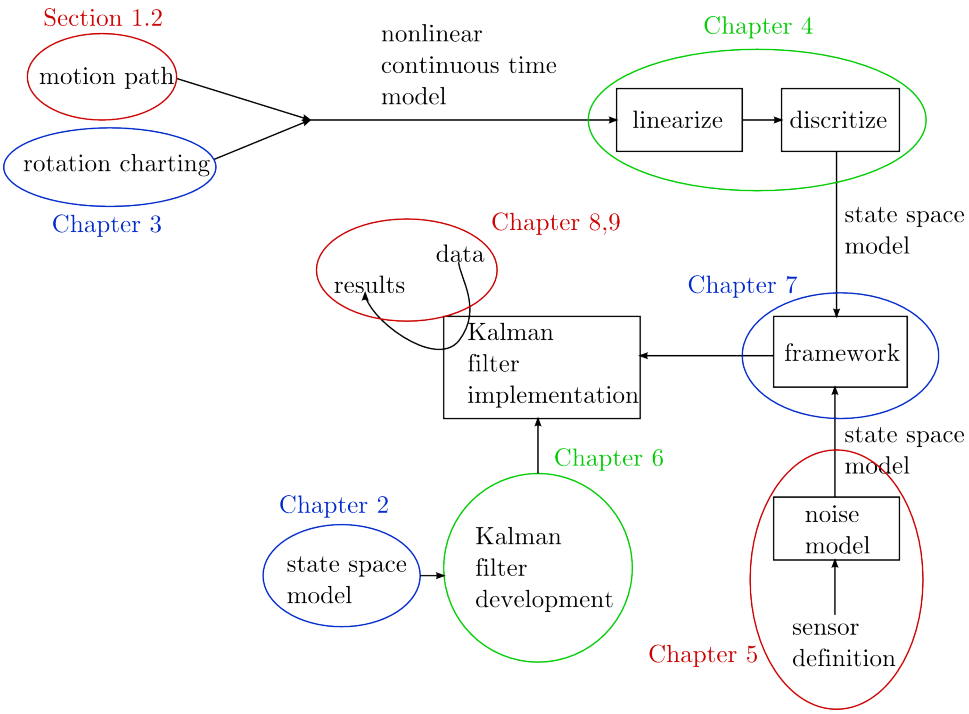

Figure 1.3: Flow of the Thesis

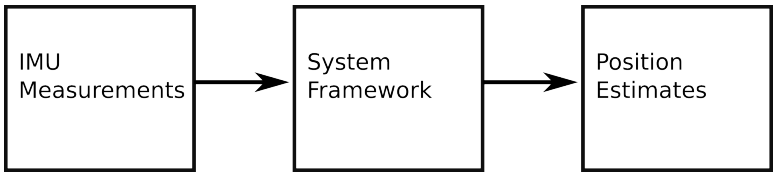

Figure 1.4: Basic Design Goal

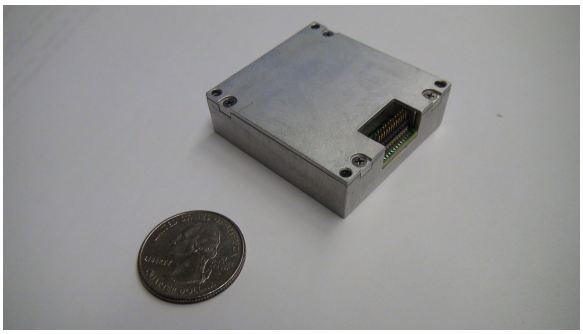

Figure 1.5: The Analog Devices ADIS16375 Inertial Measurement Unit

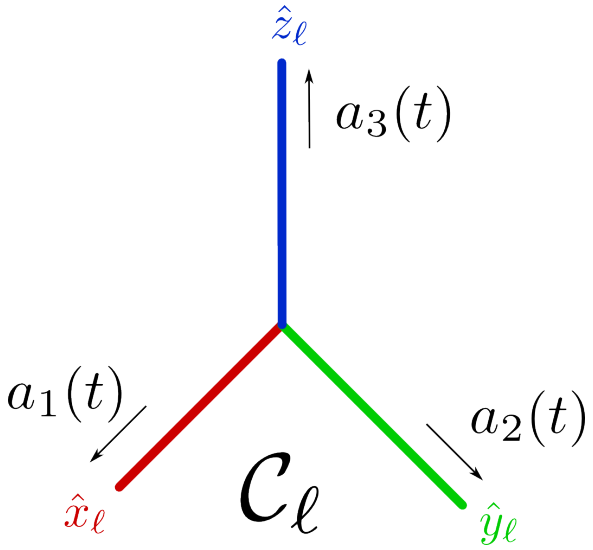

Figure 1.6: Local Accelerations

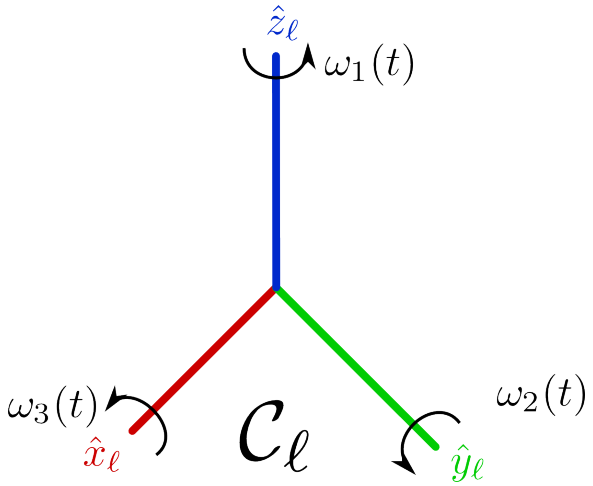

Figure 1.7: Local Rotational Velocities

Figure 1.8: Example of MEMS IMU in Foot Mounted Strap Down Configuration Shown here Attached to the RF Location Unit

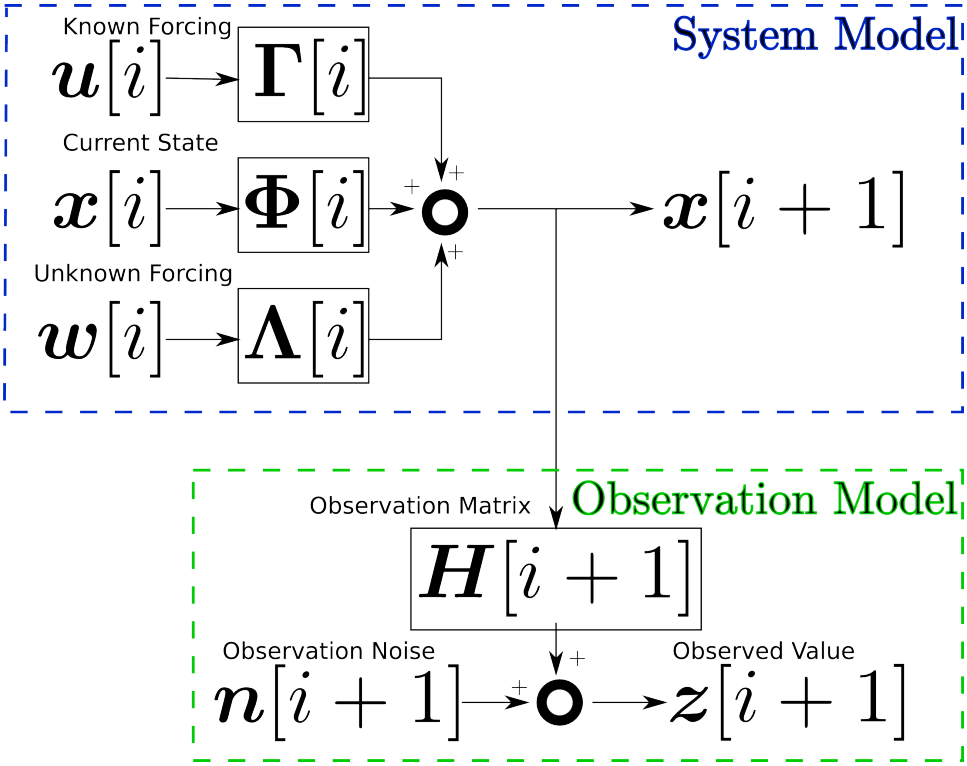

Figure 1.9: State Space Model

Ordinary Variable	x
Estimate	$\tilde{x}$
Mean	$\bar{x}$
Time Derivative	$\dot{x}$
Subindexed	$\breve{x}$
Unit Axis	$\hat{x}$

Table 1.3: Table of Variable Decorations

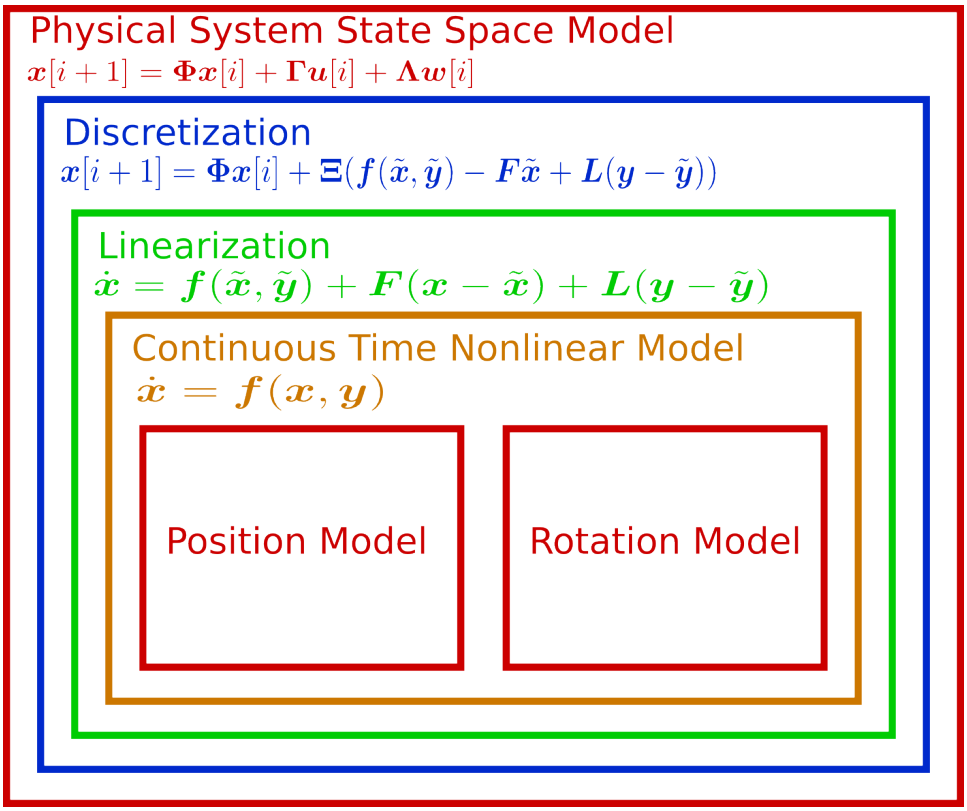

Figure 1.10: Physical System's State Space Model Breakup

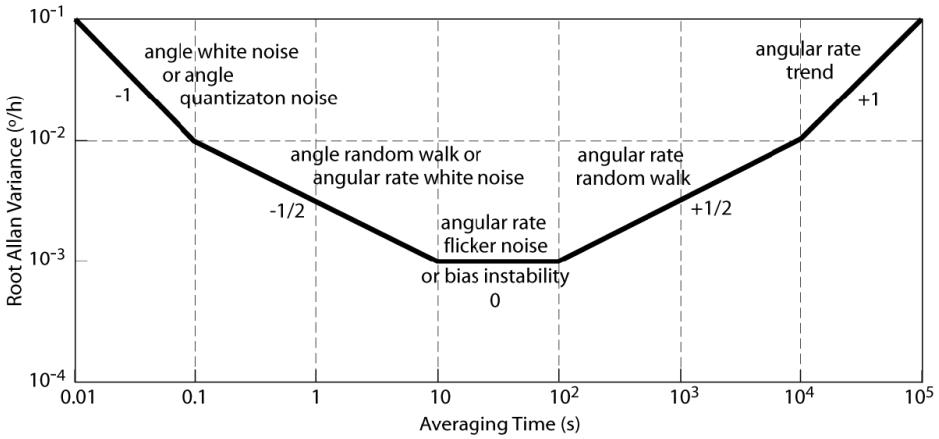

Figure 1.11: Components of IMU Sensor Noise Base on Allan Variance from ieeestd

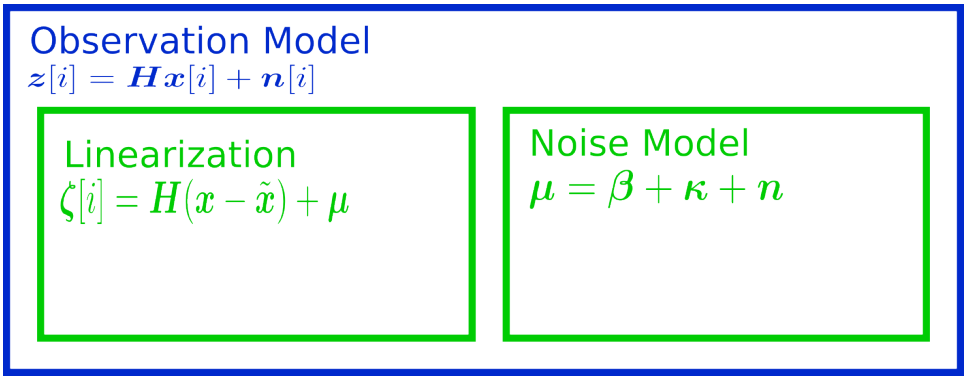

Figure 1.12: Observation Model

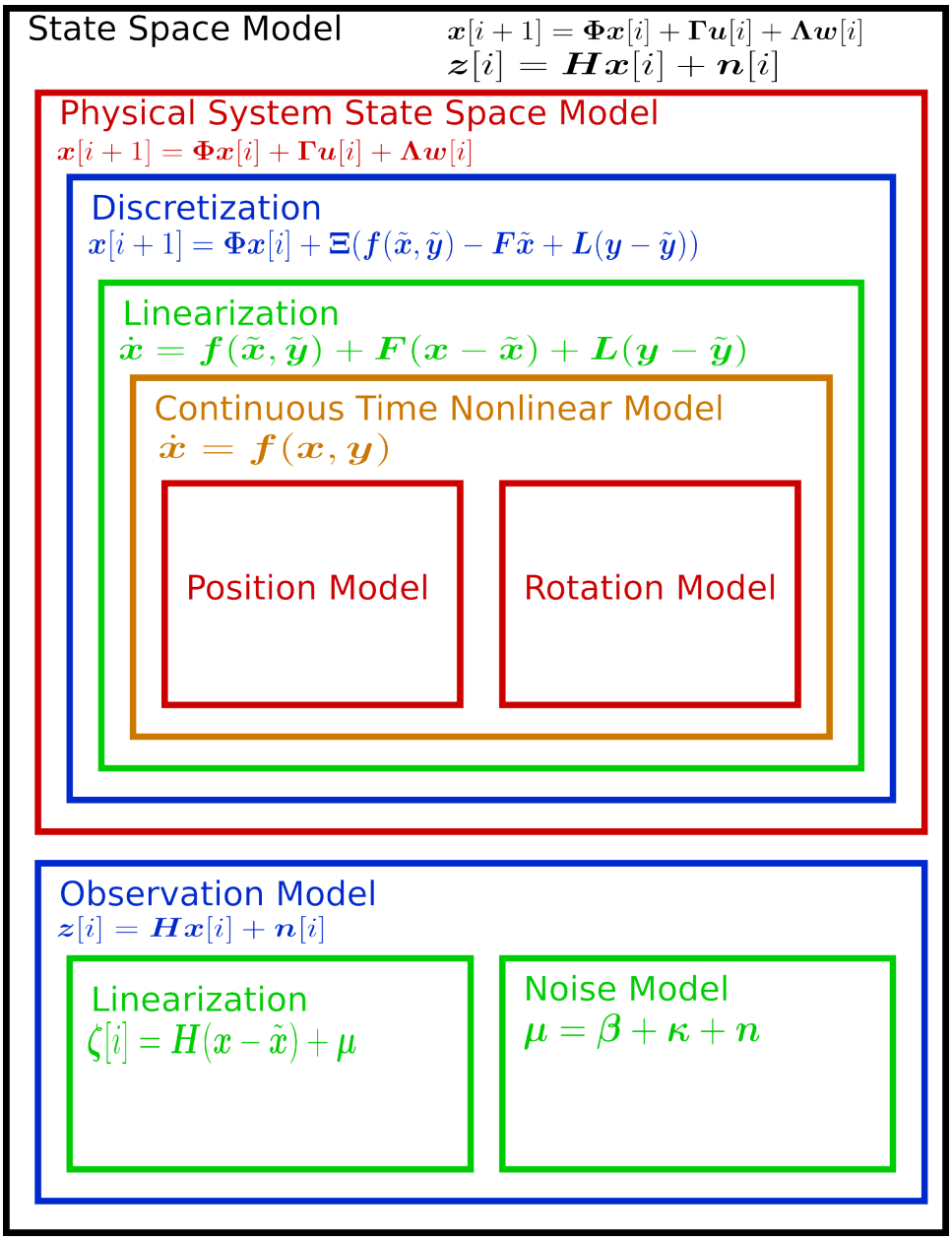

Figure 1.13: Visualization of Total System

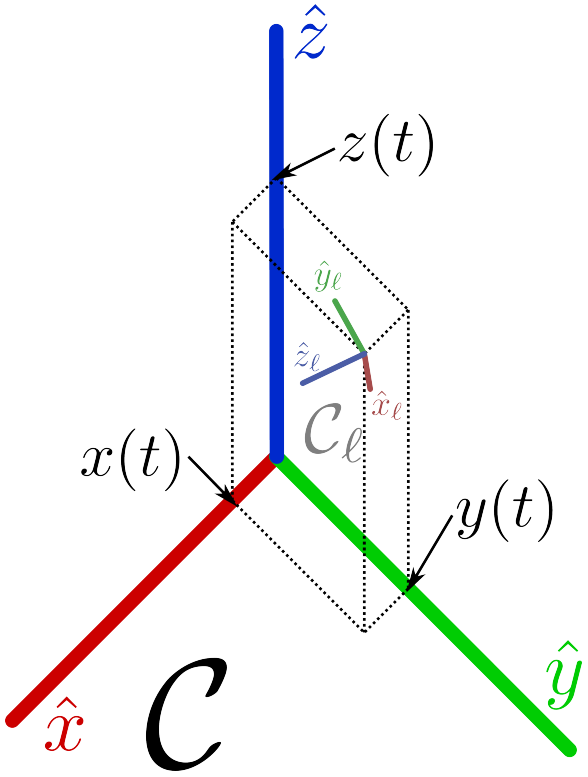

Figure 1.14: Coordinate Frames

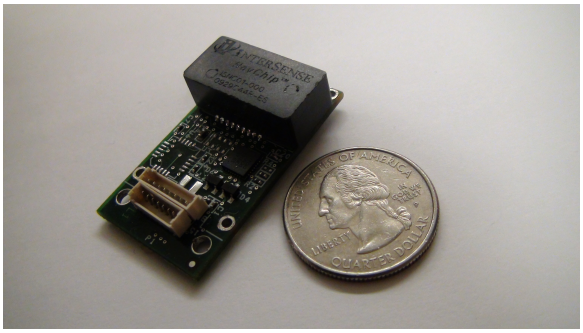

Figure 1.15: The Navchip sensor

Chapter 2

State Space Model

In order to process the data provided by the IMU we will need to use some sort of technique for the integration of the navigation equations and estimation of position, orientation, and sensor errors. Data processing is an incredibly large field with an almost innumerable number of such techniques. For our discussion we will be using the Kalman Filter, one of the most commonly employed methods implemented within the area of IMU based localization. For an example of such a system see [9]. The Kalman Filter uses a linear discrete state space model which can be written in the following form:

$$i = 0, \cdots, \eta \tag{2.1a}$$

$$\boldsymbol{x}[i+1] = \boldsymbol{\Phi}[i]\boldsymbol{x}[i] + \boldsymbol{\Gamma}[i]\boldsymbol{u}[i] + \boldsymbol{\Lambda}[i]\boldsymbol{w}[i] \tag{2.1b}$$

$$\boldsymbol{z}[i+1] = \boldsymbol{H}[i+1]\boldsymbol{x}[i+1] + \boldsymbol{n}[i+1] \tag{2.1c}$$

$$\boldsymbol{x}[0] \in \mathbb{R}^k \tag{2.1d}$$

$$\boldsymbol{x}[i+1] \in \mathbb{R}^k \tag{2.1e}$$

$$\boldsymbol{\Phi}[i] \in \mathbb{R}^{k \times k} \tag{2.1f}$$

$$\boldsymbol{\Gamma}[i] \in \mathbb{R}^{k \times j} \tag{2.1g}$$

$$\boldsymbol{u}[i] \in \mathbb{R}^j \tag{2.1h}$$

$$\boldsymbol{\Lambda}[i] \in \mathbb{R}^{k \times l} \tag{2.1i}$$

$$\boldsymbol{w}[i] \in \mathbb{R}^l \tag{2.1j}$$

$$\boldsymbol{z}[i+1] \in \mathbb{R}^m \tag{2.1k}$$

$$\boldsymbol{H}[i+1] \in \mathbb{R}^{m \times k} \tag{2.1l}$$

$$\boldsymbol{n}[i+1] \in \mathbb{R}^m \tag{2.1m}$$

$$E(\boldsymbol{x}[0]) = \bar{\boldsymbol{x}}[0] \tag{2.1n}$$

$$E\big((\boldsymbol{x}[0] - \bar{\boldsymbol{x}}[0])\,(\boldsymbol{x}[0] - \bar{\boldsymbol{x}}[0])^{\mathrm{T}}\big) = \boldsymbol{P}_0 \tag{2.1o}$$

$$E(\boldsymbol{w}[i]) = \boldsymbol{0} \tag{2.1p}$$

$$E\big(\boldsymbol{w}[i]\boldsymbol{w}[i]^{\mathrm{T}}\big) = \boldsymbol{Q}[i] \tag{2.1q}$$

$$E(\boldsymbol{n}[i+1]) = \boldsymbol{0} \tag{2.1r}$$

$$E\big(\boldsymbol{n}[i+1]\boldsymbol{n}[i+1]^{\mathrm{T}}\big) = \boldsymbol{R}[i+1] \tag{2.1s}$$

Where $\boldsymbol{x}[0]$, $\boldsymbol{w}[i]$, and $\boldsymbol{n}[i+1]$ are unknown White Gaussian Noise (WGN) processes while the functions $\boldsymbol{z}$, $\boldsymbol{u}$, $\boldsymbol{\Phi}$, $\boldsymbol{\Gamma}$, $\boldsymbol{\Lambda}$, $\boldsymbol{H}$, $\boldsymbol{Q}$, $\boldsymbol{R}$, along with the mean initial state $\bar{\boldsymbol{x}}[0]$ and its covariance matrix $\boldsymbol{P}_0$, are all known.

2.1 Two Dimensional Example

To capture the general idea of what is being suggested by this model we can start with a two dimensional system. The matrices for this example will be as follows,

$$\begin{aligned}
\bar{\boldsymbol{x}}[0] &= \begin{pmatrix} 1 \\ -1 \end{pmatrix} \\
\boldsymbol{P}_0 &= \begin{pmatrix} 2 & 1 \\ 1 & 3 \end{pmatrix} \\
\boldsymbol{\Phi} &= \begin{pmatrix} 1 & \frac{1}{2} \\ 0 & 1 \end{pmatrix} \\
\boldsymbol{\Gamma u} &= \begin{pmatrix} -1 \\ -1 \end{pmatrix} \\
\boldsymbol{\Lambda} &= \boldsymbol{I} \\
\boldsymbol{Q} &= \begin{pmatrix} 1 & 0 \\ 0 & \frac{1}{2} \end{pmatrix}.
\end{aligned}$$

As the different components of the system come into play both the probability density function, what is known about the system, and an example realization, that is, a single point associated with the density function, will be shown. For this example the initial state of the system is estimated to be $\bar{\boldsymbol{x}}[0]$ with the covariance matrix $\boldsymbol{P}_0$, but the actual realization is at the point $[1.5, -0.5]$. The example's $\boldsymbol{x}[0]$ probability density function, shown as a sequence of three contours of equal probability, and true value, a single realization of this random variable, are shown in Figure 2.1.

The system then goes through an update step, from Eq. (2.1b), which is made up of three parts. They are, a homogeneous update, $\boldsymbol{\Phi x}$, and two inhomogeneous displacements, one due to the known forcing function, $\boldsymbol{\Gamma u}$, and the other due to an unknown forcing function, $\boldsymbol{\Lambda w}$. The first, the homogeneous update, expresses how the system naturally evolves and can be seen in Figure 2.2. While the actual system is moved to a new point both the mean and the shape of the covariance of our knowledge about the system change through this

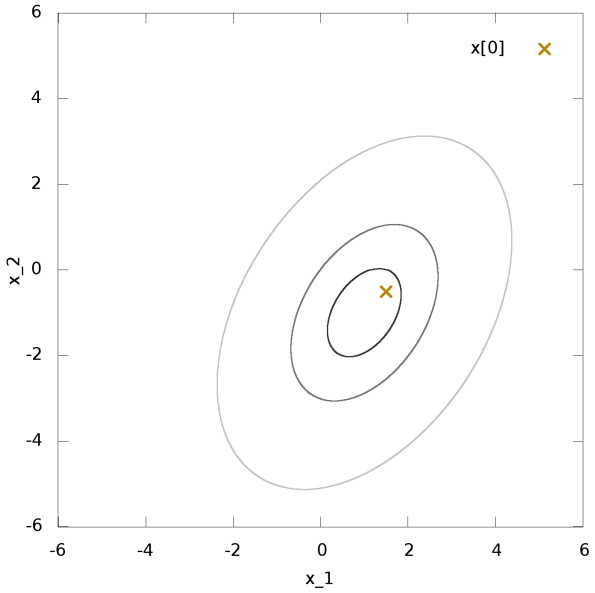

Figure 2.1: Initial Value of $\boldsymbol{x}$ at time 0

part of the update. The new values are,

$$\bar{\boldsymbol{x}}[1] = \begin{pmatrix} \frac{1}{2} \\ -1 \end{pmatrix}$$

$$E\big((\boldsymbol{x}[1] - \bar{\boldsymbol{x}}[1])\,(\boldsymbol{x}[1] - \bar{\boldsymbol{x}}[1])^{\mathrm{T}}\big) = \begin{pmatrix} 3\frac{3}{4} & 2\frac{1}{2} \\ 2\frac{1}{2} & 3 \end{pmatrix}$$

$$\boldsymbol{x}[1] = \begin{pmatrix} 1\frac{1}{4} \\ -\frac{1}{2} \end{pmatrix}.$$

This update can also be seen in the following diagram.

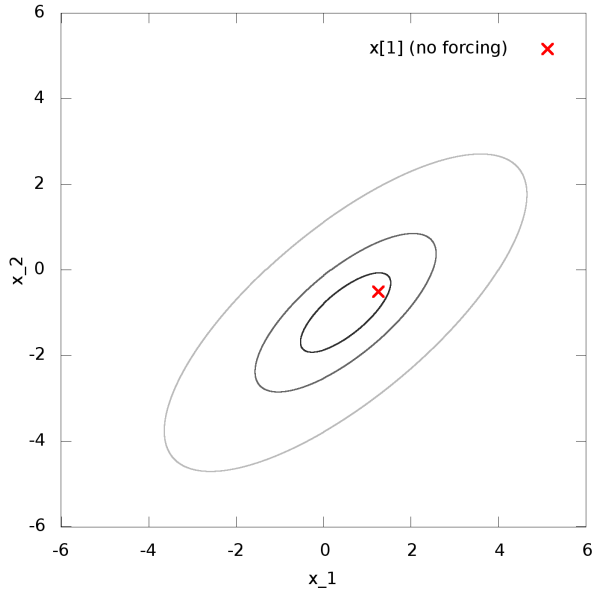

Figure 2.2: Value of $\boldsymbol{x}$ at time 1 with no forcing (Homogeneous)

Figure 2.3: Homogeneous Update

The next part of the update is the addition of a known forcing term $\boldsymbol{\Gamma u}$, here $[-1, -1]$. This force is known so it changes both the mean and the actual state by the same amount and as there is no uncertainty, it therefore does not effect the covariance. The results of our

update can be seen in Figure 2.4. The new values are,

$$\bar{\boldsymbol{x}}[1] = \begin{pmatrix} -\frac{1}{2} \\ -2 \end{pmatrix}$$

$$E\big((\boldsymbol{x}[1] - \bar{\boldsymbol{x}}[1])\,(\boldsymbol{x}[1] - \bar{\boldsymbol{x}}[1])^{\mathrm{T}}\big) = \begin{pmatrix} 3\frac{3}{4} & 2\frac{1}{2} \\ 2\frac{1}{2} & 3 \end{pmatrix}$$

$$\boldsymbol{x}[1] = \begin{pmatrix} \frac{1}{4} \\ -1\frac{1}{2} \end{pmatrix}.$$

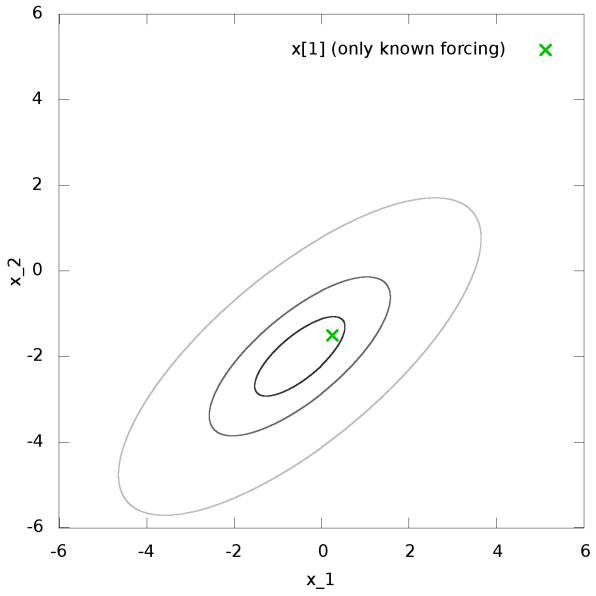

Figure 2.4: Value of $\boldsymbol{x}$ at time 1 with known forcing

This update is included in the following diagram.

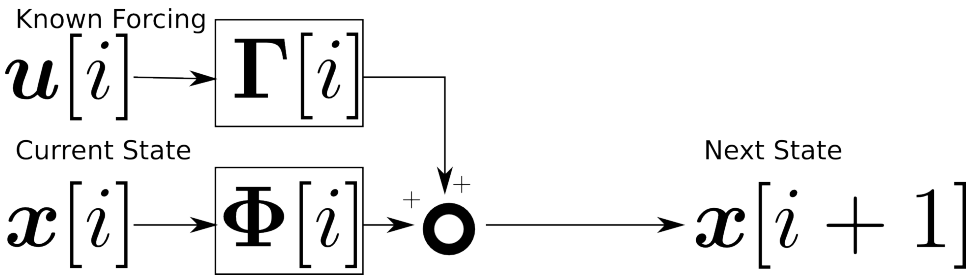

Figure 2.5: Update with Known Forcing

The system is also moved by another forcing function, this time the unknown forcing term $\boldsymbol{\Lambda w}$. As shown in Figure 2.6 we know it to have zero mean and the covariance matrix of $\boldsymbol{Q}$ but it has a realization, here $\left[0, \frac{1}{2}\right]$. This pushes the state of the system in some direction, but with only its covariance known we can only add it to the current uncertainty. This can be seen in Figure 2.7.

$$\begin{aligned} \bar{\boldsymbol{x}}[1] &= \begin{pmatrix} -\frac{1}{2} \\ -2 \end{pmatrix} \\ E\left(\left(\boldsymbol{x}[1] - \bar{\boldsymbol{x}}[1]\right)\left(\boldsymbol{x}[1] - \bar{\boldsymbol{x}}[1]\right)^{\mathrm{T}}\right) &= \begin{pmatrix} 4\frac{3}{4} & 2\frac{1}{2} \\ 2\frac{1}{2} & 3\frac{1}{2} \end{pmatrix} \\ \boldsymbol{x}[1] &= \begin{pmatrix} \frac{1}{4} \\ -1\frac{1}{2} \end{pmatrix}. \end{aligned}$$

This update is included in the following diagram.

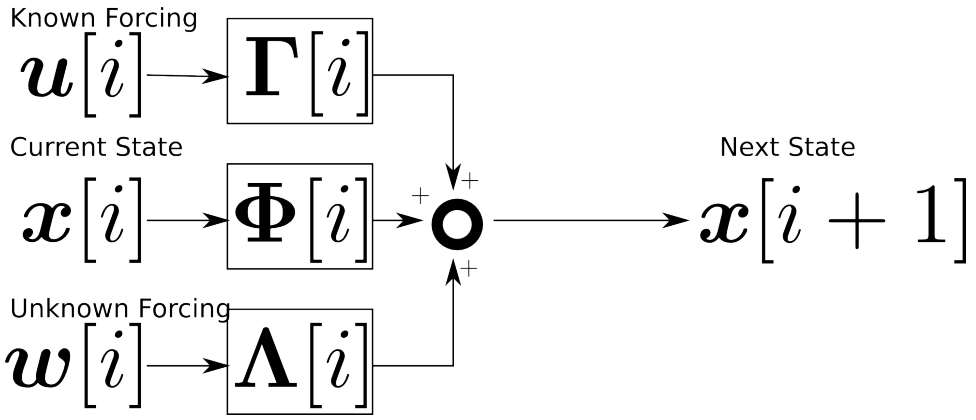

Figure 2.8: Update with Forcing

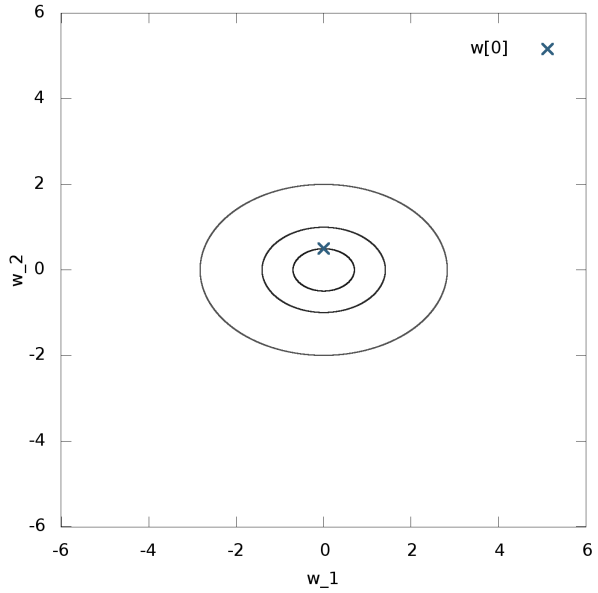

Figure 2.6: Unknown Forcing

After the system has been updated we make a measurement, from Eq. (2.1c), of the *actual* state of the system, here both x_1 and x_2. This measurement is corrupted by some unknown noise taken here to have a covariance matrix $\begin{pmatrix} 1 & 0 \\ 0 & 8 \end{pmatrix}$. This is shown in Figure 2.9.

This update and the observation model are shown in the following diagram.

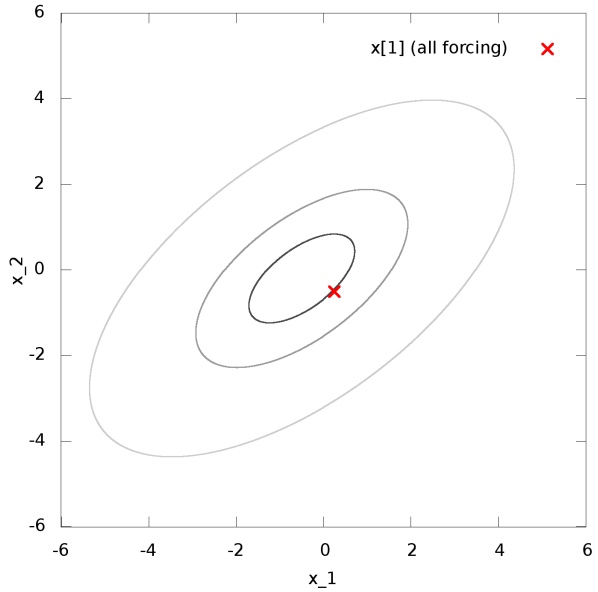

Figure 2.7: Value of $\boldsymbol{x}$ at time 1 (all forcing)

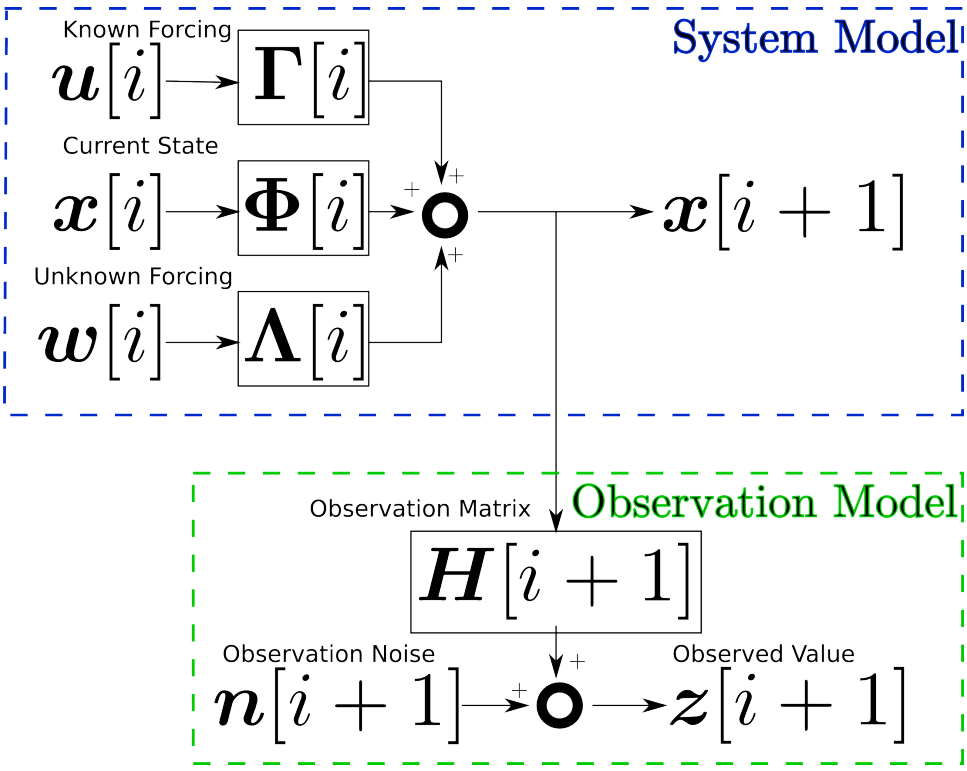

Figure 2.10: Update and Observation Model

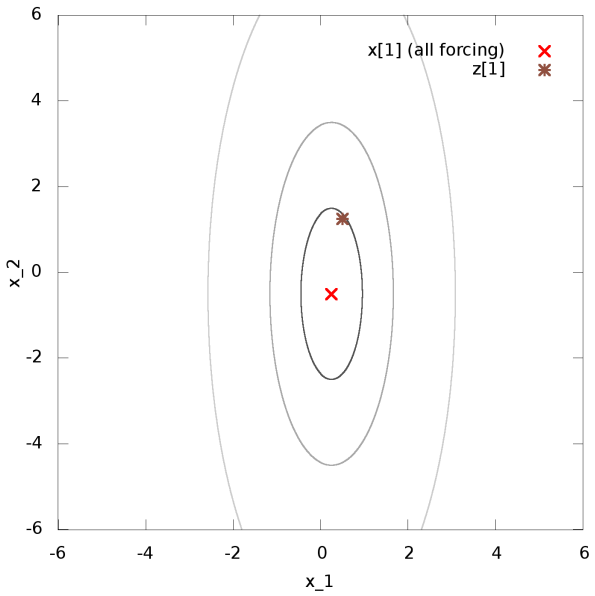

Figure 2.9: Observation, $\boldsymbol{z}$ at time 1

Chapter 3

Quaternions and their applications to the rotation tracking problem

Our description of the motions of the body from Section 1.2 included a rotation operator. In order to construct a state space system we will need to represent this rotation as a vector. To do this it will be necessary to take a short look at rotations to understand why this is perhaps one of the most important adaptations we will have to make to accomodate our system to the linear vector space structure of the state space model needed for Kalman filter processing. This problem, of representing a group with elements of $\mathbb{R}^n$ is called charting.

3.1 Rotations

A rotation in 3 dimensions (3D) is a 3D linear transform and, therefore, has a 3×3 matrix associated with it. These matrices form what is called the Special Orthogonal Group, $SO(3)$, a subgroup of all orthogonal 3×3 matrices, the Orthogonal Group, $O(3)$. An orthogonal matrix is one which possesses an inverse given by its transposes.

$$\boldsymbol{A}^{-1} = \boldsymbol{A}^{\mathrm{T}}$$

$O(3)$ is a group closed under matrix multiplication.

$$\text{If } \boldsymbol{A}, \boldsymbol{B} \in O(3)$$

$$\text{Then } \boldsymbol{A} \cdot \boldsymbol{B} \in O(3)$$

Additionally, $O(3)$ is a Lie Group within which we can find a connected subgroup, the matrices with determinant 1, $SO(3)$. $SO(3)$ is also a Lie Group with an associated Lie Algebra, $\mathfrak{so}(3)$, related to the group through exponentiation. $\mathfrak{so}(3)$ can be represented by the set of all 3×3 matrices which are skew symmetric, that is their transpose is their additive inverse.

$$\boldsymbol{A}^{\mathrm{T}} = -\boldsymbol{A}$$

The exponential map is the standard matrix one.

$$\exp(\boldsymbol{A}) = I + \sum_{n=1}^{\infty} \frac{\boldsymbol{A}^n}{n!}$$

The Lie Algebra is a group under the bracket product,

$$\boldsymbol{A} \in \mathfrak{so}(3), \boldsymbol{B} \in \mathfrak{so}(3)$$

$$[\boldsymbol{A}, \boldsymbol{B}] \in \mathfrak{so}(3),$$

where the bracket product is uniquely defined to be,

$$[\boldsymbol{A}, \boldsymbol{B}] = \boldsymbol{A} \cdot \boldsymbol{B} - \boldsymbol{B} \cdot \boldsymbol{A}.$$

If $\boldsymbol{S}(t)$ is a differentiable rotation function then its derivative, $\dot{\boldsymbol{S}}(t)$, is an element of $\mathfrak{so}(3)$. In this work we will be using Rotation Quaternions to solve the problem of charting the rotations in such a way as to achieve simplest implementation.

3.2 Quaternions

Quaternions are elements of $\mathbb{R}^4$, Rotational Quaternions, which we will be using, are those which are on the unit sphere, that is they have length 1 [12]. Quaternions form a double cover for $SO(3)$ and are what is called the Spin Group, S^3. They are a double cover because for every rotation in $SO(3)$ there is a 'positive' and 'negative' rotation quaternion. Quaternions have additional properties which we will not be using. For their use in our problem we will need a way to rotate a 3-space vector as implied by an element of the Rotation Quaternions and a differential relationship similar to Eq. (1.4).

Though not expressly needed, it will also be useful to be able to transform a 3×3 matrix representation of a rotation into a quaternion. So given a rotation matrix, $\boldsymbol{S} =$

$\begin{pmatrix} S_{11} & S_{12} & S_{13} \\ S_{21} & S_{22} & S_{23} \\ S_{31} & S_{32} & S_{33} \end{pmatrix}$, one of the two quaternions in the double cover, $\boldsymbol{q} = (q_0, q_1, q_2, q_3)$, can be found by the following equation from [12].

$$q_0 = \frac{1}{2}\sqrt{S_{11} + S_{22} + S_{33} + 1} \tag{3.1a}$$

$$q_1 = \frac{S_{23} - S_{32}}{4q_0} \tag{3.1b}$$

$$q_2 = \frac{S_{31} - S_{13}}{4q_0} \tag{3.1c}$$

$$q_3 = \frac{S_{12} - S_{21}}{4q_0} \tag{3.1d}$$

Given a quaternion $\boldsymbol{q} = (q_0, q_1, q_2, q_3)$ we can transform it back into the rotation matrix with the following formula, also from [12].

$$\boldsymbol{\mathcal{S}}(\boldsymbol{q}) = 2\begin{pmatrix} q_0^2 + q_1^2 - \frac{1}{2} & q_1q_2 + q_0q_3 & q_1q_3 - q_0q_2 \\ q_1q_2 - q_0q_3 & q_0^2 + q_2^2 - \frac{1}{2} & q_2q_3 + q_0q_1 \\ q_1q_3 + q_0q_2 & q_2q_3 - q_0q_1 & q_0^2 + q_3^2 - \frac{1}{2} \end{pmatrix} \tag{3.2}$$

If $\boldsymbol{S}(t)$ is given by the differential form $\dot{\boldsymbol{S}}(t) = \begin{pmatrix} 0 & x & -y \\ -x & 0 & z \\ y & -z & 0 \end{pmatrix}\boldsymbol{S}(t)$ then the following is the quaternion differential form from [12].

$$\dot{\boldsymbol{q}}(t) = \begin{pmatrix} 0 & -x & -y & -z \\ x & 0 & z & -y \\ y & -z & 0 & x \\ z & y & -x & 0 \end{pmatrix}\boldsymbol{q}(t) \tag{3.3}$$

3.3 Advantages of the Quaternion Rotation Representation

Along with Quaternions a few other common choices for charting $SO(3)$ were examined during the early stages of this research. Amongst them Quaternions were chosen as a compromise between the problems associated with the various charting options. The charting methods originally examined were:

'Vectorizing': That is to simply treat the 9 elements of the matrix as elements of a vector in $\mathbb{R}^9$.

Quaternions: Using Rotation Quaternions as already discussed

Euler Angles: Using one of the many different Euler angle representations.

	'Vectorizing'	Quaternions	Euler Angles
Closure	$\boldsymbol{r}+\boldsymbol{w}=\tilde{\boldsymbol{S}}=\boldsymbol{U\Sigma V}^*$ $\rightarrow \boldsymbol{UV}^*$	$\boldsymbol{r}+\boldsymbol{w}=\boldsymbol{q}$ $\rightarrow \frac{\boldsymbol{q}}{\lvert\boldsymbol{q}\rvert}$	Closed
Differential Form	$\begin{pmatrix} 0 & x & -y \\ -x & 0 & z \\ y & -z & 0 \end{pmatrix} \boldsymbol{S}$	$\begin{pmatrix} 0 & -x & -y & -z \\ x & 0 & z & -y \\ y & -z & 0 & x \\ z & y & -x & 0 \end{pmatrix} \boldsymbol{q}$	$\begin{pmatrix} 1 & \sin(\phi)\tan(\theta) & \cos(\phi)\tan(\theta) \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \sin(\phi)\sec(\theta) & \cos(\phi)\sec(\theta) \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix}$

Table 3.1: Comparison of Different Charts

With these three methods there are two major difficulties.

Closure: The filter will be taking the originally charted value, $\boldsymbol{r}$, and adding updates, $\boldsymbol{w}$. The result of this, $\boldsymbol{r}+\boldsymbol{w}$, should also lie within the chart. That is the chart should be closed under the addition of elements from the vector space it lies in.

Differential Form: Converting the derivative of the rotation matrix to the derivative of the chart with the least complexity.

It is clear that adding updates to the 'Vectorization' based representation may not result in an orthogonal and unit determinant matrix, so the 'Vectorization' method will not lead to closure. In order to find the 'closest' rotation matrix to $\boldsymbol{r}+\boldsymbol{w}$ we will need to perform, what is called, the polar decomposition, requiring a singular value decomposition. Quaternions have the same issue, as only quaternions of unit length are rotation quaternions. But in order to find the closest rotation quaternion will only require dividing by the quaternion's length. Thus there is a computational benefit to using the quaternion approach over 'Vectorization'. Euler Angles are closed under the additive operator and thus have the greatest computational benefit for this situation.

We have seen the differential form for a rotation matrix and quaternions. The differential form for an example Euler angle representation can be found in [12] and is significantly more complicated due to the nonlinear functions involved in the execution.
The difficulties of the various charting methods, closure and differential form, are shown in 3.1. We will be using quaternions because, as it has been seen, although they are not always the simplest to manipulate, they are never the most difficult.

Chapter 4

Physical System Model

We have defined our physical system enough to formulate it as a state space system as defined in Eq. (2.1), with the block diagram shown in Fig. 4.1

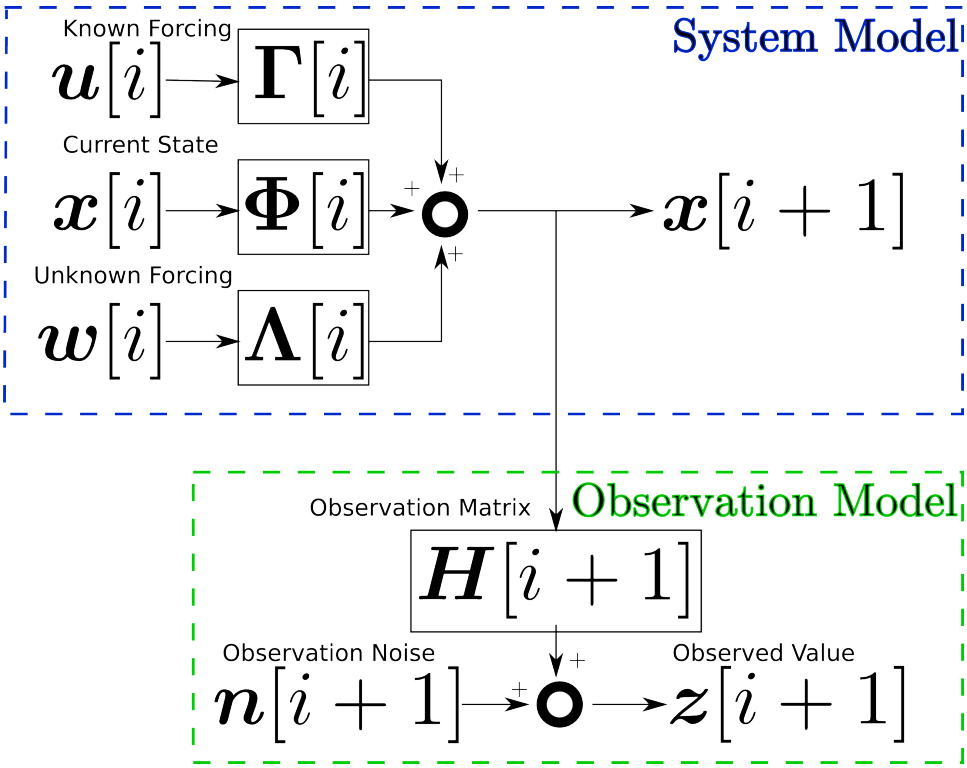

Figure 4.1: Update and Observation Model

In order to perform the various mathematical manipulations in this section we made

extensive use of the tool Maple^{TM} from MapleSoft^{TM} in order to generate both the formulas seen in this chapter and those used in other segments of the implementation code.

The breve accent, $\breve{x}$, is used here to denote that all the states defined here are only in reference to the physical system, as described in Eqs. (1.1) and (1.4), and there will be additional states defined later. Additional states used for tracking various sensor effects will be added in Chapter 5.

4.1 Nonlinear Continuous System

The current system model is a combination of equations from previous chapters.

$$\dot{p_1}(t) = v_1(t) \tag{4.1a}$$
$$\dot{p_2}(t) = v_2(t) \tag{4.1b}$$
$$\dot{p_3}(t) = v_3(t) \tag{4.1c}$$
$$\dot{v_1}(t) = a_1(t) \tag{4.1d}$$
$$\dot{v_2}(t) = a_2(t) \tag{4.1e}$$
$$\dot{v_3}(t) = a_3(t) \tag{4.1f}$$
$$\dot{a_1}(t) = j_1(t) \tag{4.1g}$$
$$\dot{a_2}(t) = j_2(t) \tag{4.1h}$$
$$\dot{a_3}(t) = j_3(t) \tag{4.1i}$$
$$\dot{g}(t) = 0 \tag{4.1j}$$
$$\dot{q_0}(t) = -1/2\omega_1(t)q_1(t) + -1/2\omega_2(t)q_2(t) + -1/2\omega_3(t)q_3(t) \tag{4.1k}$$
$$\dot{q_1}(t) = 1/2\omega_1(t)q_0(t) + 1/2\omega_3(t)q_2(t) + -1/2\omega_2(t)q_3(t) \tag{4.1l}$$
$$\dot{q_2}(t) = 1/2\omega_2(t)q_0(t) + -1/2\omega_3(t)q_1(t) + 1/2\omega_1(t)q_3(t) \tag{4.1m}$$
$$\dot{q_3}(t) = 1/2\omega_3(t)q_0(t) + 1/2\omega_2(t)q_1(t) + -1/2\omega_1(t)q_2(t) \tag{4.1n}$$
$$\dot{\omega_1}(t) = \alpha_1(t) \tag{4.1o}$$
$$\dot{\omega_2}(t) = \alpha_2(t) \tag{4.1p}$$
$$\dot{\omega_3}(t) = \alpha_3(t) \tag{4.1q}$$
$$\tag{4.1r}$$

Here $\boldsymbol{\omega}$ is the local rotational velocity, $\dot{\boldsymbol{S}}_\ell(t) = \begin{pmatrix} 0 & \omega_1(t) & -\omega_2(t) \\ -\omega_1(t) & 0 & \omega_3(t) \\ \omega_2(t) & -\omega_3(t) & 0 \end{pmatrix}$, and $\boldsymbol{\alpha}$ is the local rotational acceleration $\dot{\boldsymbol{\omega}}(t) = \boldsymbol{\alpha}(t)$. The system of equations in Eq. (4.1) can be reduced to a single vector system of differential equations.

$$\dot{\breve{\boldsymbol{x}}}(t) = \boldsymbol{f}(t, \breve{\boldsymbol{x}}(t), \boldsymbol{y}(t)),$$

where the states are

$$\breve{\boldsymbol{x}} = [p_1, p_2, p_3, v_1, v_2, v_3, a_1, a_2, a_3, g, q_0, q_1, q_2, q_3, \omega_1, \omega_2, \omega_3] \tag{4.2}$$

and the forcing functions are

$$\boldsymbol{y} = [j_1, j_2, j_3, \alpha_1, \alpha_2, \alpha_3]. \tag{4.3}$$

The system is defined with these states with the IMU's measurement capabilities in mind. The IMU measures local acceleration and local rotational velocity so acceleration and rotational velocity must appear as states in the filter. Global acceleration, instead of local acceleration, is defined because the goal of the filter is track the system in the global coordinate frame.

The rotational velocities $\boldsymbol{\omega}(t)$ are defined in relation to the local coordinate frame as shown in Fig. 4.2.

4.2 Linearization

The first step in putting this problem in the form of a state space system model is to linearize it about an assumed operating point. That is, given the estimate $[\tilde{\breve{\boldsymbol{x}}}, \tilde{\boldsymbol{y}}]$, we make the approximation

$$\dot{\breve{\boldsymbol{x}}}(t) = \boldsymbol{f}\Big(t, \tilde{\breve{\boldsymbol{x}}}(t), \tilde{\boldsymbol{y}}(t)\Big) + \boldsymbol{F} \cdot \Big(\breve{\boldsymbol{x}}(t) - \tilde{\breve{\boldsymbol{x}}}(t)\Big) + \boldsymbol{L} \cdot \Big(\boldsymbol{y}(t) - \tilde{\boldsymbol{y}}(t)\Big)$$

$$\boldsymbol{F} = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\Big(t, \tilde{\breve{\boldsymbol{x}}}(t), \tilde{\boldsymbol{y}}(t)\Big)$$

$$\boldsymbol{L} = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{y}}\Big(t, \tilde{\breve{\boldsymbol{x}}}(t), \tilde{\boldsymbol{y}}(t)\Big),$$

where $\boldsymbol{F}$ and $\boldsymbol{L}$ are the partial derivative matrices of $\boldsymbol{f}(t)$ relative to the operating point values.

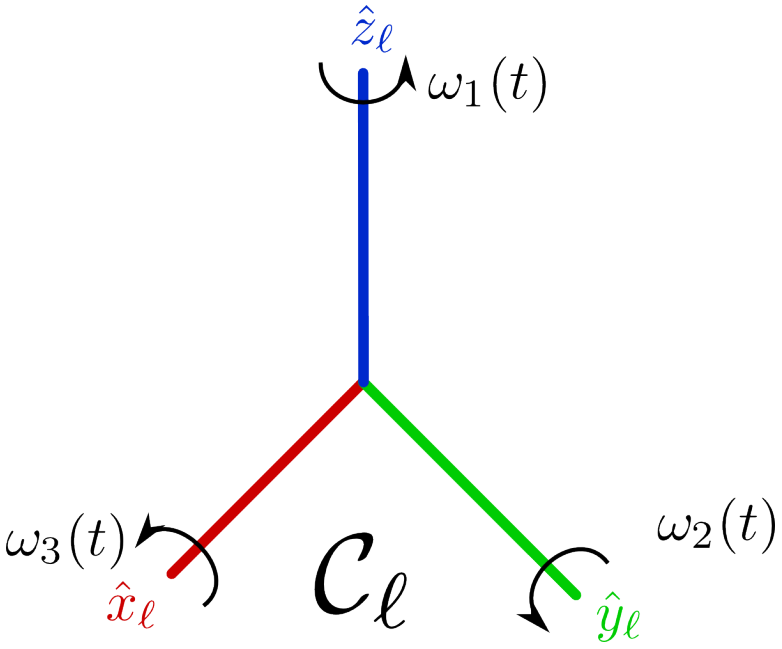

Figure 4.2: Rotational Velocities

The partial derivative matrices for our system relative to the state and forcing function

variables are given as,

$$\boldsymbol{F} = \begin{bmatrix}
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1/2\,\omega_1 & -1/2\,\omega_2 & -1/2\,\omega_3 & -1/2\,q_1 & -1/2\,q_2 & -1/2\,q_3 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1/2\,\omega_1 & 0 & 1/2\,\omega_3 & -1/2\,\omega_2 & 1/2\,q_0 & -1/2\,q_3 & 1/2\,q_2 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1/2\,\omega_2 & -1/2\,\omega_3 & 0 & 1/2\,\omega_1 & 1/2\,q_3 & 1/2\,q_0 & -1/2\,q_1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1/2\,\omega_3 & 1/2\,\omega_2 & -1/2\,\omega_1 & 0 & -1/2\,q_2 & 1/2\,q_1 & 1/2\,q_0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}, \tag{4.4}$$

$$\boldsymbol{L} = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}. \tag{4.5}$$

4.3 Discretization

The system is now prepared to be discretized on the intervals $[t_i, t_{i+1}]$. For simplicity we will assume that the matrices $\boldsymbol{F}$, $\boldsymbol{L}$, and the vectors $\boldsymbol{f}\left(t, \tilde{\tilde{\boldsymbol{x}}}(t), \tilde{\boldsymbol{y}}(t)\right)$, $\tilde{\tilde{\boldsymbol{x}}}(t)$, $\tilde{\boldsymbol{y}}(t)$, $\boldsymbol{y}(t)$ are constant on the interval. This continuous system will be discetized to obtain a sampled-time representation given by $\boldsymbol{F}[i]$, $\boldsymbol{L}[i]$, $\boldsymbol{f}[i]$, $\tilde{\tilde{\boldsymbol{x}}}[i]$, $\tilde{\boldsymbol{y}}[i]$, $\boldsymbol{y}[i]$ respectively. The discrete values of $\boldsymbol{x}[i]$ will evolve according to the difference function

$$\breve{\boldsymbol{x}}[i+1] = \breve{\boldsymbol{\Phi}}[i]\breve{\boldsymbol{x}}[i] + \boldsymbol{\Xi}[i]\left(\boldsymbol{f}[i] - \boldsymbol{F}[i]\tilde{\tilde{\boldsymbol{x}}}[i]\right) + \boldsymbol{\Xi}[i]\boldsymbol{L}[i](\boldsymbol{y}[i] - \tilde{\boldsymbol{y}}[i]).$$

From pages 79-85 of [16] we have the following formula for calculating the matrices $\breve{\boldsymbol{\Phi}}[i]$ and $\boldsymbol{\Xi}[i]$, assuming the forcing functions are constant across the time increment.

$$\epsilon[i] = t_{i+1} - t_i$$

$$\breve{\boldsymbol{\Phi}}[i] = \exp\left(\boldsymbol{F}[i]\epsilon[i]\right)$$

$$\boldsymbol{\Xi}[i] = \breve{\boldsymbol{\Phi}}[i] \int_0^{\epsilon[i]} \exp\left(-\boldsymbol{F}[i]\tau\right) d\tau$$

For numerical reasons we will be approximating these matrices with their Taylor series of order 1. For our system $\epsilon \sim \frac{1}{100}$ so $\epsilon^2 \sim \frac{1}{10000}$ and the tracked object will never be moving fast enough for this or future terms to have a substantial impact.

$$\breve{\boldsymbol{\Phi}}[i] \approx \boldsymbol{I} + \boldsymbol{F}[i]\epsilon[i]$$

$$\boldsymbol{\Xi}[i] \approx \boldsymbol{I}\epsilon[i]$$

In our case

$$\breve{\boldsymbol{\Phi}} = \begin{bmatrix}
1 & 0 & 0 & \epsilon & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & \epsilon & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & \epsilon & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & \epsilon & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & \epsilon & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & \epsilon & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1/2\,\epsilon\,\omega_1 & -1/2\,\epsilon\,\omega_2 & -1/2\,\epsilon\,\omega_3 & -1/2\,\epsilon\,q_1 & -1/2\,\epsilon\,q_2 & -1/2\,\epsilon\,q_3 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1/2\,\epsilon\,\omega_1 & 1 & 1/2\,\epsilon\,\omega_3 & -1/2\,\epsilon\,\omega_2 & 1/2\,\epsilon\,q_0 & -1/2\,\epsilon\,q_3 & 1/2\,\epsilon\,q_2 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1/2\,\epsilon\,\omega_2 & -1/2\,\epsilon\,\omega_3 & 1 & 1/2\,\epsilon\,\omega_1 & 1/2\,\epsilon\,q_3 & 1/2\,\epsilon\,q_0 & -1/2\,\epsilon\,q_1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1/2\,\epsilon\,\omega_3 & 1/2\,\epsilon\,\omega_2 & -1/2\,\epsilon\,\omega_1 & 1 & -1/2\,\epsilon\,q_2 & 1/2\,\epsilon\,q_1 & 1/2\,\epsilon\,q_0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}. \tag{4.6}$$

4.4 Gaussian Uncertainty

If we assume that $\breve{\boldsymbol{x}}[0]$ is a multivariate Gaussian random vector and that

$$E(\breve{\boldsymbol{x}}[0]) = \bar{\breve{\boldsymbol{x}}}[0]$$
$$E\Big(\big(\breve{\boldsymbol{x}}[0] - \bar{\breve{\boldsymbol{x}}}[0]\big)\big(\breve{\boldsymbol{x}}[0] - \bar{\breve{\boldsymbol{x}}}[0]\big)^{\mathrm{T}}\Big) = \breve{\boldsymbol{P}}_0$$

and that $\boldsymbol{y}[i]$ is also a multivariate Gaussian random vector

$$\begin{aligned} E(\boldsymbol{y}[i]) &= \tilde{\boldsymbol{y}}[i] \\ \breve{\boldsymbol{w}}[i] &= \boldsymbol{y}[i] - \tilde{\boldsymbol{y}}[i] \\ E\big(\breve{\boldsymbol{w}}[i]\breve{\boldsymbol{w}}[i]^{\mathrm{T}}\big) &= \breve{\boldsymbol{Q}}[i] \end{aligned} \tag{4.7}$$

Then we may rewrite our system as

$$\breve{\boldsymbol{x}}[i+1] = \breve{\boldsymbol{\Phi}}[i]\breve{\boldsymbol{x}}[i] + \breve{\boldsymbol{\Gamma}}[i]\breve{\boldsymbol{u}}[i] + \breve{\boldsymbol{\Lambda}}[i]\breve{\boldsymbol{w}}[i], \tag{4.8}$$

where the new matrices are defined

$$\breve{\boldsymbol{\Gamma}}[i] = \begin{pmatrix} \boldsymbol{\Xi}[i] & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{\Xi}[i]\boldsymbol{F}[i] \end{pmatrix} \tag{4.9}$$

$$\breve{\boldsymbol{\Lambda}}[i] = \boldsymbol{\Xi}[i]\boldsymbol{L}[i] \tag{4.10}$$

and the forcing function $\breve{\boldsymbol{u}}[i]$ is defined

$$\breve{\boldsymbol{u}}[i] = \left[\boldsymbol{f}[i], -\tilde{\tilde{\boldsymbol{x}}}[i]\right]. \tag{4.11}$$

We have the beginnings of a state space system as described in (2.1).

4.5 Nonlinear Observations

Any observations of global frame's variables made within the local coordinate frame, specifically those made by the IMU, will be nonlinear, as they observe the product of the current rotation and global frame's variables. In general we have a nonlinear observation corrupted by WGN $\boldsymbol{n}[i]$,

$$\boldsymbol{\zeta}[i] = \boldsymbol{g}[i, \boldsymbol{x}[i]] + \boldsymbol{n}[i].$$

Using our approximate value for $\boldsymbol{x}$, $\tilde{\boldsymbol{x}}$, we can make the linear approximation,

$$\boldsymbol{\zeta}[i] = \boldsymbol{g}[i, \boldsymbol{x}[i]] + \boldsymbol{H}[i] \cdot \left(\boldsymbol{x}[i] - \tilde{\boldsymbol{x}}[i]\right) + \boldsymbol{n}[i],$$

where

$$\boldsymbol{H}[i] = \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{x}}[i, \tilde{\boldsymbol{x}}[i]] \tag{4.12}$$

Reorganizing this to become

$$\boldsymbol{\zeta}[i] - \boldsymbol{g}[i, \tilde{\boldsymbol{x}}[i]] + \boldsymbol{H}[i]\tilde{\boldsymbol{x}}[i] = \boldsymbol{H}[i]\boldsymbol{x}[i] + \boldsymbol{n}[i].$$

we can recognize the form for $\boldsymbol{z}[i]$ from Eq. (2.1c) and define it to be

$$\boldsymbol{z}[i] = \boldsymbol{\zeta}[i] - \boldsymbol{g}[i, \tilde{\boldsymbol{x}}[i]] + \boldsymbol{H}[i]\tilde{\boldsymbol{x}}[i] \tag{4.13}$$

We now have the Linear Observation equation from (2.1c)

The result of the Chapter can be seen in the following block diagram.

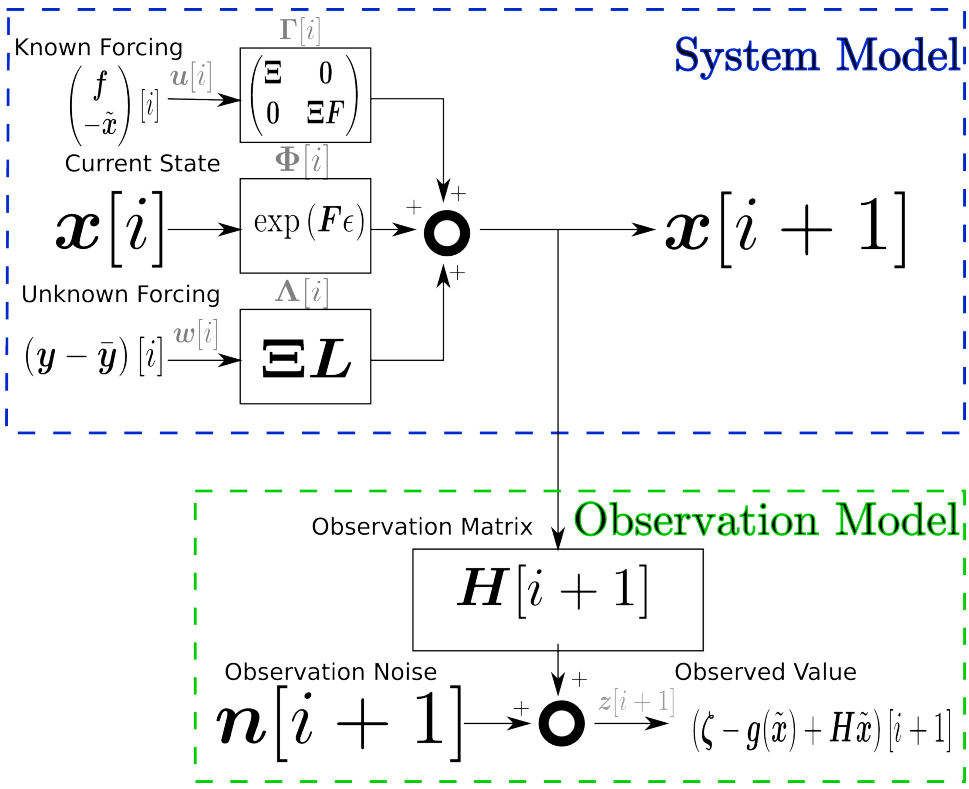

Figure 4.3: Linearized and Discritized System with Nonlinear Observation

Chapter 5

Modeling Sensor Noise

Up until this point we have been discussing the physical system we wish to model. Recall, from (1.3), that the IMU measures the physical properties of the system with additional noise. It is the goal of this chapter to discuss noise found on each of the signals from the 3 accelerometers and 3 gyroscopes of the IMU and a strategy of how to model it. The strategy for constructing a noise model is taken from Section 19.6 of [1].

5.1 Allan Variance

The statistic of interest for the IMU's sensors, usually provided by the manufactures' data sheets, is Allan Variance. The Allan Variance of a signal $x(t)$ for a given averaging interval ϵ is given by (5.1)

$$\sigma^2(x,\epsilon) = \frac{1}{2\epsilon^2} E\left(\left(\int_0^{\epsilon} x(s)ds - \int_{\epsilon}^{2\epsilon} x(s)ds\right)^2\right) \tag{5.1}$$

When the square root of this quantity, the Allan Deviation, is plotted on a log-log scale various types of noise can be isolated and estimated. These types are determined by their slope in the log-log Allan Deviation plot, per the recommendations of [1], and are named in Table 5.1 along with a set of estimated values for an inertial sensor unit from the Intersense company, the Navchip isnc01-000.

5.2 Our Sensor

The specific IMU we used in this work, the Navchip isnc01-000 made by InterSense Inc. [10], has 3 accelerometers and 3 gyroscopes. For each of these we will construct a noise model.

5.2.1 Gyroscope Noise

From the data sheet we can find the specified Allan Deviation plot, shown in Figure 5.1[1]. From this plot we can estimate the three dominant noise sources in the sensor using a straight edge.

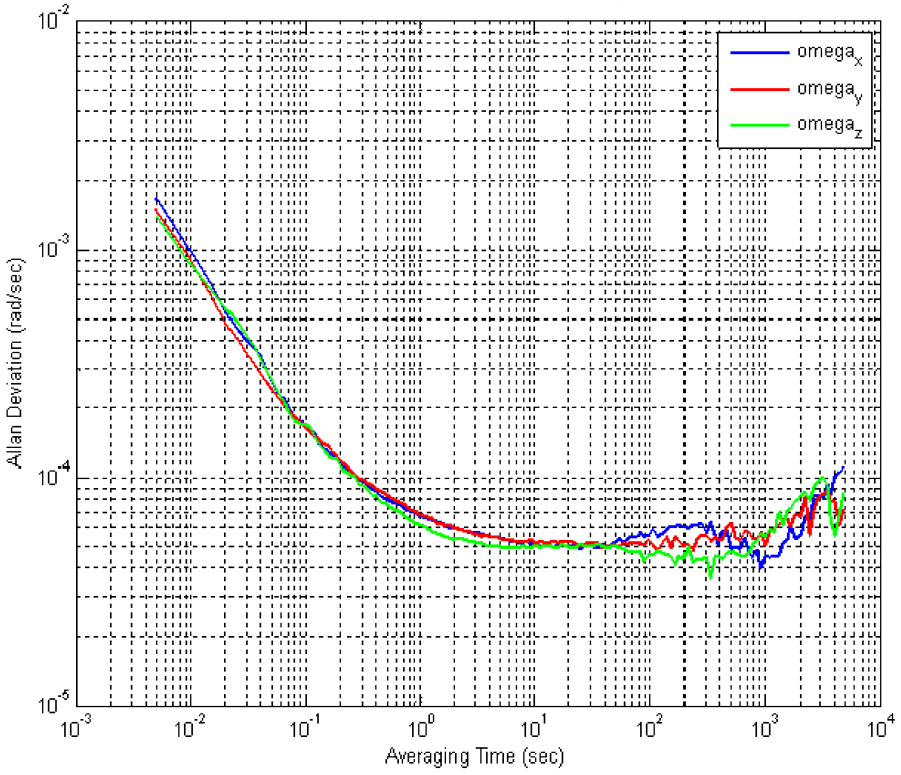

Figure 5.1: Navchip isnc01-000 gyroscope Allan Deviation

Our model for the noise is three distinct stochastic processes,

$$\check{z}[i] = \chi[i] + \beta[i] + \kappa[i] + \check{n}[i], \tag{5.2}$$

[1]This and other information about the Navchip isnc01-000 used with the permission of the Intersense company

	Gaussian Noise	Flicker Noise	Bias Walk
log-log Allan Deviation Slope	$-\frac{1}{2}$	0	$\frac{1}{2}$
Estimated 1s Value $\frac{\text{rad}}{\text{s}}$	$.5e-4$	$.5e-4$	$.12e-5$

Table 5.1: Table of Gyroscope Allan Deviation Values and estimated values for a Navchip isnc01-000

where $\breve{n}$ is the Gaussian noise, β is the bias walk, κ is the flicker noise, χ is the desired quantity, and z is the measurement returned by the sensor.

Starting with the Gaussian noise we can identify the specific models for each of these processes and their associated WGN variances. The Gaussian noise will be a WGN process defined in (5.3).

$$E(\breve{n}[i]) = 0 \tag{5.3a}$$

$$E\left(\breve{n}[i]^2\right) = \frac{q^2}{\epsilon} \tag{5.3b}$$

$$\sigma(\breve{n}, \epsilon) = \frac{q}{\sqrt{2\epsilon}} \tag{5.3c}$$

Here σ is the square root of the Allan variance, the Allan deviation, found in Table 5.1. We can plug in $\epsilon = 1$ s and solving for q, the deviation of the WGN process, $q = .7071067810e-4$.

This noise is purely Gaussian and will be the $\breve{n}$ term of observation equation (2.1c).

Similarly we may model the bias walk.

$$\breve{\beta}[i+1] = \breve{\beta}[i] + y[i] \tag{5.4a}$$

$$E(y[i]) = 0 \qquad y[i] \text{ GWN} \tag{5.4b}$$

$$E\left(y[i]^2\right) = b^2\epsilon \tag{5.4c}$$

$$\sigma\left(\breve{\beta}, \epsilon\right) = \frac{b\sqrt{\epsilon}}{\sqrt{2}} \tag{5.4d}$$

Again by plugging in $\epsilon = 1$ we can solve for b, the deviation of the WGN process which is driving the bias walk, $b = .1697056274e-5$. This noise source, $\breve{\beta}$, will have to be tracked as an additional state in our system.

The final noise source, flicker noise, is more complicated. For this particular noise we will not be formulating it as a single noise source, but as the combination of multiple *colored*

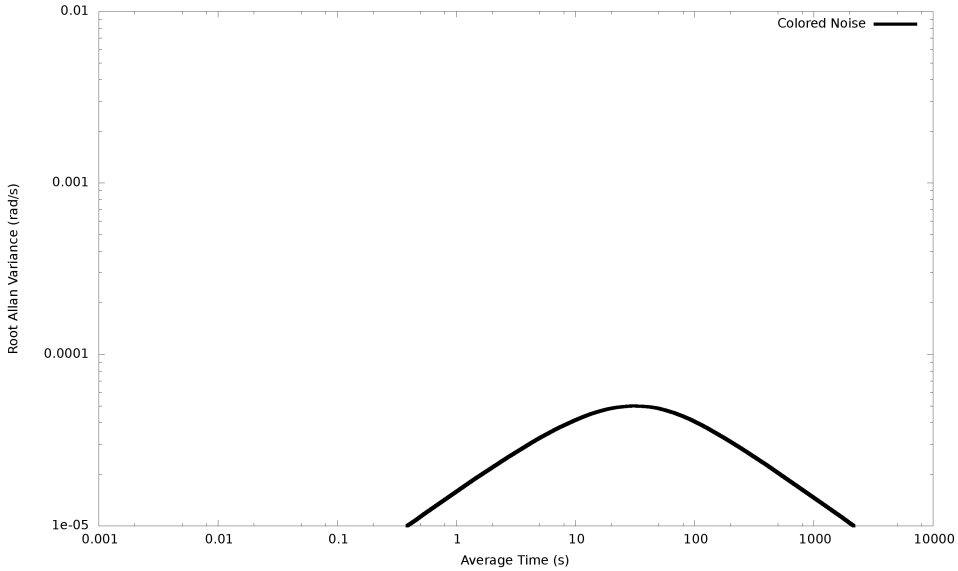

Figure 5.2: Sample Colored Noise Allan Deviation

noise sources. A colored noise source is defined in Eq. (5.5) in accord with the development on page 123 of [11].

$$\dot{m}(t) = -c_1 m(t) + c_1 c_2 p(t) \tag{5.5a}$$

$$p \text{ is WGN autocovariance } \delta(t) \tag{5.5b}$$

$$E(m(t)m(\tau)) = \frac{c_1 c_2^2}{2} \exp\left(-c_1 \left|t-\tau\right|\right) \tag{5.5c}$$

$$\sigma(m,\epsilon) = 1/2\,\sqrt{2}\sqrt{\frac{{c_2}^2\left(-3+4\,\mathrm{e}^{-c_1\epsilon}+2\,c_1\epsilon-\mathrm{e}^{-2\,c_1\epsilon}\right)}{c_1}}\epsilon^{-1} \tag{5.5d}$$

An example of the Allan deviation log-log plot for colored noise with values $c_1 = 6E-2$, $c_2 = 5E-4$ is shown in Fig. 5.2.

For the Gyros we will be approximating the flicker region, from 1 s to 1000 s as estimated with a straight edge, with a colored noise processes placed evenly in the region. Like the bias walk noise, each colored noise processes will need its own state in the system. The final

system repesenting the gyroscope measurements is

$$\breve{\boldsymbol{x}}[i+1] = \begin{bmatrix} 1 & 0 \\ 0 & \mathrm{e}^{-0.05984800888\,\epsilon} \end{bmatrix} \breve{\boldsymbol{x}}[i] + \breve{\boldsymbol{w}}[i] \tag{5.6a}$$

$$E(\breve{\boldsymbol{w}}[i]\breve{\boldsymbol{w}}[i]) = \begin{bmatrix} 2.879999997 \times 10^{-12}\,\epsilon^{-1} & 0 \\ 0 & 0.0000000007851212103\,\epsilon \end{bmatrix} \tag{5.6b}$$

$$\breve{z}[i] = \omega + \begin{bmatrix} 1 & 1 \end{bmatrix} \breve{\boldsymbol{x}}[i] + \breve{n} \tag{5.6c}$$

$$E\left(\breve{n}[i]^2\right) = 0.000000002499999998\,\epsilon^{-1} \tag{5.6d}$$

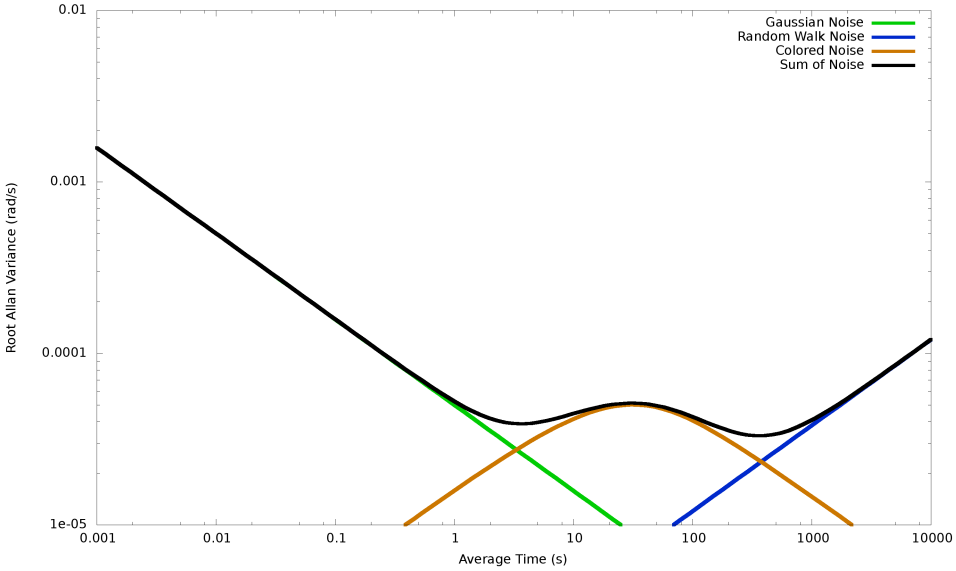

Figure 5.3: Simulation model for gyroscope derived in Section 5.2.1 shown in the form of a Noise Allan Deviation Plot

All of the noise processes can be seen plotted together in Figure 5.3 with the resulting combined process Allan Deviation Plot.

	Gaussian Noise	Flicker Noise	Bias Walk	Colored Noise
log-log Allan Deviation Slope	$-\frac{1}{2}$	0	$\frac{1}{2}$	
X Estimated 1s Value $\frac{m/s^2}{s}$	$.49e-3$	$.22e-3$	$.39e-4$	0
Y Estimated 1s Value $\frac{m/s^2}{s}$	$.49e-3$	$.22e-3$	$.78e-4$	0
Z Estimated 1s Value $\frac{m/s^2}{s}$	$.49e-3$	$.22e-3$	$.11e-4$	1

Table 5.2: Table of Accelerometer Allan Deviation Values

5.2.2 Accelerometer Noise

In the same way as the gyroscope sensor's noise model was developed in the last section, we may generate a noise model for the accelerometers.

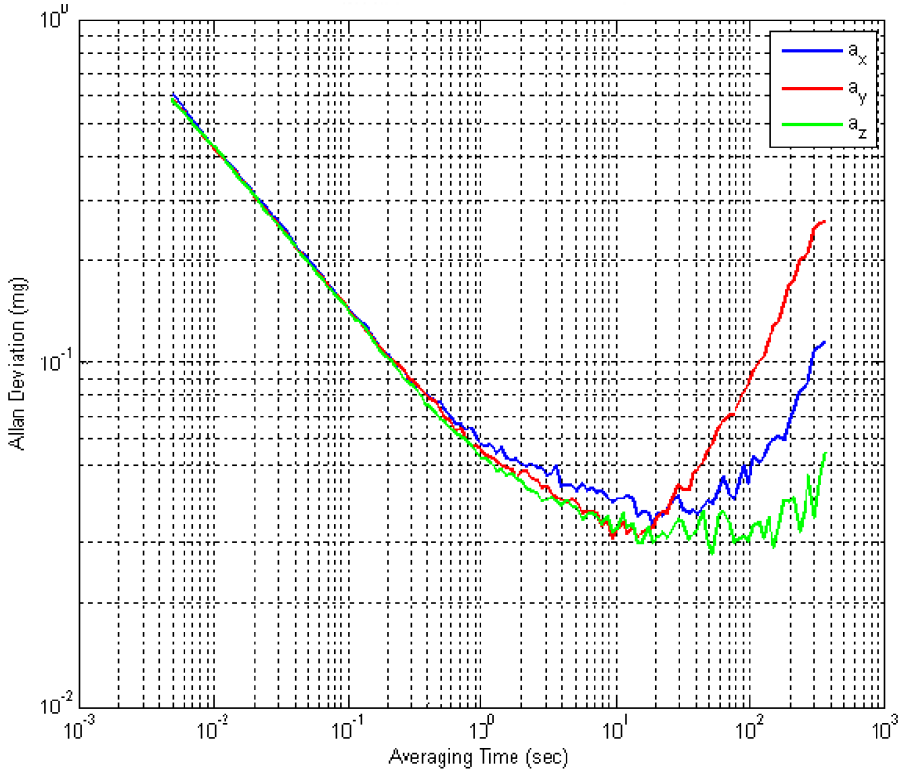

Figure 5.4: Navchip isnc01-000 accelerometer Allan Deviation

As can be seen in Figure 5.4 each of the accelerometers is significantly different and each is going to need its own specific model. These models may designed just as the gyroscope

noise model was with the values specified in Table 5.2. The models for the X and Y accelerometers can be approximated without the addition of colored noise models because there is a sufficiently small flat region in their characteristic curves.
The resulting models are shown in equations (5.7), (5.8), and (5.9)

$$\breve{\boldsymbol{x}}[i+1] = \begin{bmatrix} 1 \end{bmatrix} \breve{\boldsymbol{x}}[i] + \breve{\boldsymbol{w}}[i] \tag{5.7a}$$

$$E(\breve{\boldsymbol{w}}[i]\breve{\boldsymbol{w}}[i]) = \begin{bmatrix} 0.000000003079555198\,\epsilon^{-1} \end{bmatrix} \tag{5.7b}$$

$$\breve{z}[i] = a + \begin{bmatrix} 1 \end{bmatrix} \breve{\boldsymbol{x}}[i] + \breve{n} \tag{5.7c}$$

$$E\left(\breve{n}[i]^2\right) = 0.0000002405902499\,\epsilon^{-1} \tag{5.7d}$$

$$\breve{\boldsymbol{x}}[i+1] = \begin{bmatrix} 1 \end{bmatrix} \breve{\boldsymbol{x}}[i] + \breve{\boldsymbol{w}}[i] \tag{5.8a}$$

$$E(\breve{\boldsymbol{w}}[i]\breve{\boldsymbol{w}}[i]) = \begin{bmatrix} 0.000000012318220 78\,\epsilon^{-1} \end{bmatrix} \tag{5.8b}$$

$$\breve{z}[i] = a + \begin{bmatrix} 1 \end{bmatrix} \breve{\boldsymbol{x}}[i] + \breve{n} \tag{5.8c}$$

$$E\left(\breve{n}[i]^2\right) = 0.0000002405902499\,\epsilon^{-1} \tag{5.8d}$$

$$\breve{\boldsymbol{x}}[i+1] = \begin{bmatrix} 1 & 0 \\ 0 & \mathrm{e}^{-0.04886703347\,\epsilon} \end{bmatrix} \breve{\boldsymbol{x}}[i] + \breve{\boldsymbol{w}}[i] \tag{5.9a}$$

$$E(\breve{\boldsymbol{w}}[i]\breve{\boldsymbol{w}}[i]) = \begin{bmatrix} 0.00000000023289136 20\,\epsilon^{-1} & 0 \\ 0 & 0.000000011943905 19\,\epsilon \end{bmatrix} \tag{5.9b}$$

$$\breve{z}[i] = a + \begin{bmatrix} 1 & 1 \end{bmatrix} \breve{\boldsymbol{x}}[i] + \breve{n} \tag{5.9c}$$

$$E\left(\breve{n}[i]^2\right) = 0.0000002405902499\,\epsilon^{-1} \tag{5.9d}$$

The resulting model's Allan Deviations can be seen in Figure 5.5

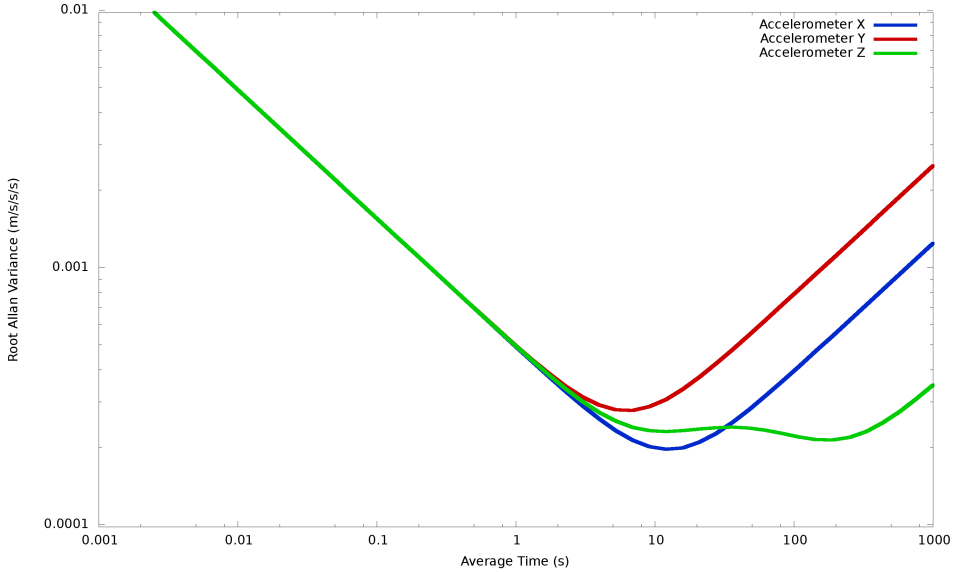

Figure 5.5: Simulation Model for Accelerometer Derived in Section 5.2.2 shown in the form of a Noise Allan Deviation Plot

Chapter 6

Kalman Filter

We will work here to present a discussion of the Kalman Filter from an understanding of Optimal Control Theory. This work, though self standing can be a little oblique at times; for reference see Appendix A for a very similar construction in continuous time, where some concepts are more readily available. This development is believed to present a novel perspective of the Kalman Filter Derivation.

6.1 Problem Description

Consider a state space system as described in (2.1) with the additional requirement that $\boldsymbol{\Phi}[i]$ be invertible, that is that the system is reversible. Given an initial condition, $\boldsymbol{x}[0]$, and the pair of inputs, $(\boldsymbol{w}, \boldsymbol{n})$, the system is fully determined with $\boldsymbol{x}$ defined, leaving the identity $\boldsymbol{n}[i+1] = \boldsymbol{z}[i+1] - \boldsymbol{H}[i+1]\boldsymbol{x}[i+1]$. Consider the probability densities of each of the variables defined as follows.

$$p(\boldsymbol{x}[0]) = \frac{1}{(2\pi)^{\frac{k}{2}} |\boldsymbol{P}_0|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}\left(\boldsymbol{x}[0] - \bar{\boldsymbol{x}}[0]\right)^{\mathrm{T}} \boldsymbol{P}_0^{-1} \left(\boldsymbol{x}[0] - \bar{\boldsymbol{x}}[0]\right)\right)$$

$$\mathfrak{w}(\boldsymbol{w}) = \prod_{i=0}^{\eta} \frac{1}{(2\pi)^{\frac{l}{2}} |\boldsymbol{Q}[i]|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}\boldsymbol{w}[i]^{\mathrm{T}} \boldsymbol{Q}[i]^{-1} \boldsymbol{w}[i]\right)$$

$$\mathfrak{n}(\boldsymbol{n}) = \prod_{i=0}^{\eta} \frac{1}{(2\pi)^{\frac{m}{2}} |\boldsymbol{R}[i+1]|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}\boldsymbol{n}[i+1]^{\mathrm{T}} \boldsymbol{R}[i+1]^{-1} \boldsymbol{n}[i+1]\right)$$

These random variables are independent so the joint probability density is simply the product of the marginal densities.

$$\mathfrak{p}(\boldsymbol{x}[0], \boldsymbol{w}, \boldsymbol{n}) = p(\boldsymbol{x}[0])\mathfrak{w}(\boldsymbol{w})\mathfrak{n}(\boldsymbol{n})$$

Our goal, then, is to maximize this joint probability, to find the most likely series of events. By making the substitution $\boldsymbol{n}[i+1] = \boldsymbol{z}[i+1] - \boldsymbol{H}[i+1]\boldsymbol{x}[i+1]$ we may reformulate our problem of maximizing $\mathfrak{p}$ into a problem of minimizing (6.1) by applying a logarithmic transformation of the joint probability and eliminating constants which do not contribute to the minimization.

$$\begin{aligned}\mathfrak{c}(\boldsymbol{w},\boldsymbol{x}) = &\left(\boldsymbol{x}[0] - \bar{\boldsymbol{x}}[0]\right)^{\mathrm{T}} \boldsymbol{P}_0^{-1} \left(\boldsymbol{x}[0] - \bar{\boldsymbol{x}}_0\right) \\ &+ \sum_{i=0}^{\eta} \boldsymbol{w}[i]^{\mathrm{T}} \boldsymbol{Q}[i]^{-1} \boldsymbol{w}[i] \\ &+ \sum_{s=1}^{\eta+1} \left(\boldsymbol{z}[s] - \boldsymbol{H}[s]\boldsymbol{x}[s]\right)^{\mathrm{T}} \boldsymbol{R}[s]^{-1} \left(\boldsymbol{z}[s] - \boldsymbol{H}[s]\boldsymbol{x}[s]\right)\end{aligned} \tag{6.1}$$

The system's dynamics from (2.1b) can be written as constraints in the form

$$F(\boldsymbol{w},\boldsymbol{x})_{y,i+1} = -x[i+1]_y + \boldsymbol{\Phi}[i]_y \boldsymbol{x}[i] + \boldsymbol{\Gamma}[i]_y \boldsymbol{u}[i] + \boldsymbol{\Lambda}[n]_y \boldsymbol{w}[i] = 0 \tag{6.2}$$

We have now phrased our problem in the context of Convex Analysis and can use the conditions for an optimal solution of an *ordinary convex program* from Theorem 28.3 on page 281 of [15].

$$\boldsymbol{0} \in \{\partial\mathfrak{c} + \Upsilon_{1,1}\partial F_{1,1} + \cdots + \Upsilon_{y,i+1}\partial F_{y,i+1} + \cdots \Upsilon_{k,\eta+1}\partial F_{k,\eta+1}\} \tag{6.3}$$

Here ∂ indicates the subdifferential, an extension of the concept of differentiation for convex functions. The subdifferential is a set valued, function but here, because our functions are continuous, it is simply the partial derivative in terms of all inputs.
$\boldsymbol{\Upsilon} \in \mathbb{R}^{k\times\eta+1}$ is called a Kuhn-Tucker vector, though in this context, where constraints are all equalities, is is also more commonly known as the vector of Lagrangian multipliers. Instead of finding a solution which minimizes $\mathfrak{c}$ and meets the constraints of $\boldsymbol{F}$, we must find a solution which meets both the constraints of $\boldsymbol{F}$, Eq. (6.2), and a Kuhn-Tucker vector, $\boldsymbol{\Upsilon}$, that meets the condition of Eq. (6.3).

6.2 Solution

To find the solution to the problem we begin by expanding the partial derivatives of $\mathfrak{c}$ so we can illuminate the structure of the result,

$$\partial\mathfrak{c} = \left[\frac{\partial\mathfrak{c}}{\partial\boldsymbol{x}[0]}, \frac{\partial\mathfrak{c}}{\partial\boldsymbol{x}[1]}, \cdots, \frac{\partial\mathfrak{c}}{\partial\boldsymbol{x}[i]}, \cdots, \frac{\partial\mathfrak{c}}{\partial\boldsymbol{x}[\eta+1]}, \frac{\partial\mathfrak{c}}{\partial\boldsymbol{w}[1]}, \cdots, \frac{\partial\mathfrak{c}}{\partial\boldsymbol{w}[i]}, \cdots, \frac{\partial\mathfrak{c}}{\partial\boldsymbol{w}[\eta]}\right].$$

This is a vector in $\mathbb{R}^{k(\eta+1)+l\eta}$. Each component is expanded below.

$$\begin{aligned}
\frac{\partial \mathfrak{c}}{\partial \boldsymbol{x}[0]} &= -2\bar{\boldsymbol{x}}[0]^{\mathrm{T}}\boldsymbol{P}_0^{-1} + 2\boldsymbol{x}[0]^{\mathrm{T}}\boldsymbol{P}_0^{-1} \\
\frac{\partial \mathfrak{c}}{\partial \boldsymbol{x}[s]} &= -2\boldsymbol{z}[s]^{\mathrm{T}}\boldsymbol{R}[s]^{-1}\boldsymbol{H}[s] + 2\boldsymbol{x}[s]^{\mathrm{T}}\boldsymbol{H}[s]^{\mathrm{T}}\boldsymbol{R}[s]^{-1}\boldsymbol{H}[s] \quad s = 1, 2, \cdots, \eta+1 \\
\frac{\partial \mathfrak{c}}{\partial \boldsymbol{w}[i]} &= 2\boldsymbol{w}[i]^{\mathrm{T}}\boldsymbol{Q}[i]^{-1} \quad i = 0, 1, \cdots, \eta
\end{aligned}$$

Similarly we may expand the constraint functions, inserting zeros where no dependence is found.

$$\partial F_{y,i+1} = \left[\cdots, 0, \frac{\partial F_{y,i+1}}{\partial \boldsymbol{x}[i]}, \frac{\partial F_{y,i+1}}{\partial \boldsymbol{x}[i+1]}, 0, \cdots, 0, \frac{\partial F_{y,i+1}}{\partial \boldsymbol{w}[i]}, 0, \cdots\right]$$

The partial derivative of the $\boldsymbol{F}$ function in terms of $\boldsymbol{x}[i+1]$ is a vector of zeros except at the k^{th} position, where it is -1.

$$\begin{aligned}
\frac{\partial F_{y,i+1}}{\partial \boldsymbol{x}[i+1]} &= (0, 0, \cdots, 0, -1, 0, \cdots 0) \\
\frac{\partial F_{y,i+1}}{\partial \boldsymbol{x}[i]} &= \boldsymbol{\Phi}[i]_y \\
\frac{\partial F_{y,i+1}}{\partial \boldsymbol{w}[i]} &= \boldsymbol{\Lambda}[i]_y
\end{aligned}$$

Multiplying everything from Eq. (6.3) by $-\frac{1}{2}$, absorbing it into $\boldsymbol{\Upsilon}$ where possible, will simplify the result, a very similar step can be found in Appendix A.

The partial derivatives are all the same length and so they sum term-wise. For example,

$$\partial \mathfrak{c} + \Upsilon_{1,1}\partial F_{1,1} = \left[\frac{\partial \mathfrak{c}}{\partial \boldsymbol{x}[0]} + \Upsilon_{1,1}\frac{\partial F_{1,1}}{\partial \boldsymbol{x}[0]}, \cdots, \frac{\partial \mathfrak{c}}{\partial \boldsymbol{w}[\eta]} + \Upsilon_{1,1}\frac{\partial F_{1,1}}{\partial \boldsymbol{w}[\eta]}\right]$$

Partial derivatives taken in terms of $\boldsymbol{x}[0]$ are

$$\begin{aligned}
\text{from } \mathfrak{c}, \ & \left(\bar{\boldsymbol{x}}[0] - \boldsymbol{x}[0]\right)^{\mathrm{T}}\boldsymbol{P}_0^{-1} \\
\text{from } F_{y,1}, \ & -\Upsilon_{y,1}\boldsymbol{\Phi}[1]_y \\
\text{otherwise}, \ & 0.
\end{aligned}$$

Partials taken in terms of $\boldsymbol{x}[r]$, $r = 1, 2, \cdots, \eta$ are

$$\begin{aligned}
\text{from } \mathfrak{c}, \ & \left(\boldsymbol{z}[r] - \boldsymbol{H}[r]\boldsymbol{x}[r]\right)^{\mathrm{T}}\boldsymbol{R}[r]^{-1}\boldsymbol{H}[r] \\
\text{from } F_{y,r}, \ & \Upsilon_{y,r}\,(0, 0, \cdots, 0, -1, 0, \cdots, 0) \\
\text{from } F_{y,r+1}, \ & \Upsilon_{y,r+1}\boldsymbol{\Phi}[r]_y.
\end{aligned}$$

Partials taken in terms of $\boldsymbol{w}[i]$, $i = 0, 1, \cdots, \eta$ are

$$\text{from } \mathfrak{c}, \ -\boldsymbol{w}[i]^{\mathrm{T}}\boldsymbol{Q}[i]^{-1}$$
$$\text{from } F_{y,i+1}, \ \Upsilon_{y,i+1}\boldsymbol{\Lambda}[i]_y.$$

From this we can solve for an optimal estimate of $\boldsymbol{w}[i]$, $\tilde{\boldsymbol{w}}[i]$, in terms of an optimal estimate of $\boldsymbol{\Upsilon}$, $\tilde{\boldsymbol{\Upsilon}}$, by examining elements of the combined vector which are dependant on $\boldsymbol{w}[i]$.
In order to satisfy Eq. (6.3)

$$-\tilde{\boldsymbol{w}}[i]^{\mathrm{T}}\boldsymbol{Q}[i]^{-1} + \sum_{y=1}^{k} \tilde{\Upsilon}_{y,i+1}\boldsymbol{\Lambda}[i]_y = \boldsymbol{0}.$$

This simplifies to

$$\tilde{\boldsymbol{w}}[i] = \boldsymbol{Q}[i]\boldsymbol{\Lambda}[i]^{\mathrm{T}}\tilde{\boldsymbol{\lambda}}[i+1],$$

where

$$\tilde{\boldsymbol{\lambda}}[s] = \left(\tilde{\Upsilon}_{1,s}, \tilde{\Upsilon}_{2,s}, \cdots, \tilde{\Upsilon}_{y,s}, \cdots, \tilde{\Upsilon}_{k,s}\right).$$

The same logic can be applied to solving for the parts created by the partial derivatives in terms of $\boldsymbol{x}[r], r = 1, 2 \cdots, \eta$ to find the optimal estimate of $\tilde{\boldsymbol{x}}[r]$.

$$\left(\boldsymbol{z}[r] - \boldsymbol{H}[r]\tilde{\boldsymbol{x}}[r]\right)^{\mathrm{T}} \boldsymbol{R}[r]^{-1}\boldsymbol{H}[r] - \tilde{\boldsymbol{\lambda}}[r] + \tilde{\boldsymbol{\lambda}}[r+1]^{\mathrm{T}}\boldsymbol{\Phi}[r] = 0$$

So $\tilde{\boldsymbol{\lambda}}[r]$ is governed by the difference equation

$$\tilde{\boldsymbol{\lambda}}[r+1] = \boldsymbol{\Phi}[r]^{-1\mathrm{T}}\left(\tilde{\boldsymbol{\lambda}}[r] - \boldsymbol{H}[r]^{\mathrm{T}}\boldsymbol{R}[i]^{-1}\left(\boldsymbol{z}[r] - \boldsymbol{H}[r]\tilde{\boldsymbol{x}}[r]\right)\right). \tag{6.4}$$

From the partial in $\boldsymbol{x}[0]$ we get the initial condition for the optimal estimates

$$\tilde{\boldsymbol{\lambda}}[0] = \boldsymbol{P}_0^{-1}\left(\tilde{\boldsymbol{x}}[0] - \bar{\boldsymbol{x}}[0]\right). \tag{6.5}$$

From the partial in $\boldsymbol{x}[\eta+1]$ we get the final state condition

$$\tilde{\boldsymbol{\lambda}}[\eta+1] = \boldsymbol{H}[\eta+1]^{\mathrm{T}}\boldsymbol{R}[\eta+1]^{-1}\left(\boldsymbol{z}[\eta+1] - \boldsymbol{H}[\eta+1]\tilde{\boldsymbol{x}}[\eta+1]\right). \tag{6.6}$$

We will define the special matrices which relate how a different initial condition $\tilde{\boldsymbol{x}}[0]$ would change the current final states of $\tilde{\boldsymbol{x}}[s]$ and $\tilde{\boldsymbol{\lambda}}[s]$,

$$\boldsymbol{\Xi}[s] = \frac{\partial\tilde{\boldsymbol{\lambda}}[s]}{\partial\tilde{\boldsymbol{x}}[0]}$$

$$\boldsymbol{\Psi}[s] = \frac{\partial \tilde{\boldsymbol{x}}[s]}{\partial \tilde{\boldsymbol{x}}[0]}.$$

These matrices obey the update equations

$$\boldsymbol{\Xi}[i+1] = \boldsymbol{\Phi}[i]^{-1\mathrm{T}} \left(\boldsymbol{\Xi}[i] + \boldsymbol{H}[i]^{\mathrm{T}} \boldsymbol{R}[i]^{-1} \boldsymbol{H}[i] \boldsymbol{\Psi}[i] \right) \tag{6.7}$$

$$\boldsymbol{\Psi}[i+1] = \boldsymbol{\Phi}[i] \boldsymbol{\Psi}[i] + \boldsymbol{\Lambda}[i] \boldsymbol{Q}[i] \boldsymbol{\Lambda}[i]^{\mathrm{T}} \boldsymbol{\Xi}[i+1], \tag{6.8}$$

with the initial conditions

$$\boldsymbol{\Xi}[0] = \boldsymbol{P}_0^{-1} \tag{6.9}$$

$$\boldsymbol{\Psi}[0] = \boldsymbol{I}. \tag{6.10}$$

Let $\underset{\sim}{\boldsymbol{x}}, \underset{\sim}{\boldsymbol{\lambda}}$ be the optimal solution to the above equations up at the time $s-1$ (the decoration under the variable is used here to differentiate it from the optimal estimate at time s). That is, the final state is, from Eq. (6.6),

$$\underset{\sim}{\boldsymbol{\lambda}}[s-1] = \boldsymbol{H}[s-1]^{\mathrm{T}} \boldsymbol{R}[s-1]^{-1} \left(\boldsymbol{z}[s-1] - \boldsymbol{H}[s-1] \underset{\sim}{\boldsymbol{x}}[s-1] \right),$$

the next step of the system will be

$$\underset{\sim}{\boldsymbol{\lambda}}[s] = 0.$$

At time s, however, when we receive a new observation we will need to change the control so that it is an optimal estimate for the time s. That is

$$\tilde{\boldsymbol{\lambda}}[s] = \boldsymbol{H}[s]^{\mathrm{T}} \boldsymbol{R}[s]^{-1} \left(\boldsymbol{z}[s] - \boldsymbol{H}[s] \tilde{\boldsymbol{x}}[s] \right) \tag{6.11}$$

We will achieve this by making an adjustment to $\underset{\sim}{\boldsymbol{x}}[0]$ of $\acute{\boldsymbol{x}}_0$ and carrying its impact through to the end states.

$$\tilde{\boldsymbol{x}}[0] = \underset{\sim}{\boldsymbol{x}}[0] + \acute{\boldsymbol{x}}_0$$

$$\boldsymbol{\Xi}[s] \acute{\boldsymbol{x}}_0 = \boldsymbol{H}[s]^{\mathrm{T}} \boldsymbol{R}[s]^{-1} \left(\boldsymbol{z}[s] - \boldsymbol{H}[s] \underset{\sim}{\boldsymbol{x}}[s] - \boldsymbol{H}[s] \boldsymbol{\Psi}[s] \acute{\boldsymbol{x}}_0 \right)$$

$$\acute{\boldsymbol{x}}_0 = \left(\boldsymbol{\Xi}[s] + \boldsymbol{H}[s]^{\mathrm{T}} \boldsymbol{R}[s]^{-1} \boldsymbol{H}[s] \boldsymbol{\Psi}[s] \right)^{-1} \boldsymbol{H}[s] \boldsymbol{R}[s]^{-1} \cdot \left(\boldsymbol{z}[s] - \boldsymbol{H}[s] \underset{\sim}{\boldsymbol{x}}[s] \right)$$

Given this correction to $\tilde{\boldsymbol{x}}[0]$ we will need to adjust the previous value for $\underset{\sim}{\boldsymbol{x}}[s]$ to its corrected form $\tilde{\boldsymbol{x}}[s]$.

$$\tilde{\boldsymbol{x}}[s] = \underset{\sim}{\boldsymbol{x}}[s] + \boldsymbol{\Psi}[s] \acute{\boldsymbol{x}}_0$$

We can extract the Kalman Filter's equations by defining some additional matrices. The notation here is partially borrowed from [16].
The *covariance estimate update* is given by

$$\boldsymbol{P}(+)[i] = \boldsymbol{\Psi}[i]\left(\boldsymbol{\Xi}[i] + \boldsymbol{H}[i]^{\mathrm{T}}\boldsymbol{R}[i]^{-1}\boldsymbol{H}[i]\boldsymbol{\Psi}[i]\right)^{-1}. \tag{6.12}$$

The *covariance estimate extrapolation* is given by

$$\boldsymbol{P}(-)[i+1] = \boldsymbol{\Phi}[i+1]^{\mathrm{T}}\boldsymbol{P}(+)[i]\boldsymbol{\Phi}[i+1] + \boldsymbol{\Lambda}[i+1]\boldsymbol{Q}[i+1]\boldsymbol{\Lambda}[i+1]^{\mathrm{T}}. \tag{6.13}$$

Notice this can be expressed as

$$\boldsymbol{P}(-)[i+1] = \boldsymbol{\Psi}[i+1]\boldsymbol{\Xi}[i+1]^{-1}. \tag{6.14}$$

And the *covariance estimate update* can also be expressed by

$$\boldsymbol{P}(+)[i+1] = \left(\boldsymbol{P}(-)[i+1]^{-1} + \boldsymbol{H}[i+1]^{\mathrm{T}}\boldsymbol{R}[i+1]^{-1}\boldsymbol{H}[i+1]\right)^{-1}. \tag{6.15}$$

So then

$$\boldsymbol{\Psi}[s]\left(\boldsymbol{\Xi}[s] + \boldsymbol{H}[s]^{\mathrm{T}}\boldsymbol{R}[s]^{-1}\boldsymbol{H}[s]\boldsymbol{\Psi}[s]\right)^{-1}\boldsymbol{H}[s]\boldsymbol{R}[s]^{-1}$$

can be written as

$$\left(\boldsymbol{P}(-)[s]^{-1} + \boldsymbol{H}[s]^{\mathrm{T}}\boldsymbol{R}[s]^{-1}\boldsymbol{H}[s]\right)^{-1}\boldsymbol{H}[s]\boldsymbol{R}[s]^{-1}.$$

Using the binomial matrix inversion theorem, specifically a block matrix inversion property, this simplifies to the Kalman gain equation,

$$\boldsymbol{K}[s] = \boldsymbol{P}(-)[s]\boldsymbol{H}[s]^{\mathrm{T}}\left(\boldsymbol{H}[s]\boldsymbol{P}(-)[s]\boldsymbol{H}[s]^{\mathrm{T}} + \boldsymbol{R}[s]\right)^{-1} \tag{6.16}$$

which simplifies our update step to

$$\tilde{\boldsymbol{x}}[s] = \underset{\sim}{\boldsymbol{x}}[s] + \boldsymbol{K}[s]\Big(\boldsymbol{z}[s] - \boldsymbol{H}[s]\underset{\sim}{\boldsymbol{x}}[s]\Big). \tag{6.17}$$

6.3 Advantage of Our Derivation

The above derivation gives us a definition for the matrix $\boldsymbol{P}[s]$ which is not based on the notion of covariance but instead based on the relationship between the problem and its dual. This means we can track $\boldsymbol{P}[s]$ with the equations (6.13) and (6.15) even in the event where we use non-optimal Kalman gain matrix, or perform non-optimal updates. Also tracking

the entire system $(\boldsymbol{\lambda}[s], \boldsymbol{x}[s])$ can allow one to perform more infrequent updates as all the observation information is contained in $\boldsymbol{\lambda}$. By applying an update at the end of the interval that fixes $\boldsymbol{\lambda}$ to its optimal estimate we can correct for all the stored measurements in the interval.

The primary advantage that we will be trying to exploit is its potential to correct linearization errors in the past with future data. To see an example of how this methodology could achieve this see Appendix B.

Chapter 7

Total System Framework

It is now possible to bring the ideas discussed in the previous chapters into one framework. We have defined the components of the system from the Newtonian physical system model in Chapter 4 and the various noise models in Chapter 5. These subsystems will be merged into one total system with additional discussion provided where needed.

7.1 Total System

We define the state variables of the total system,

$$[p_1, p_2, p_3, v_1, v_2, v_3, a_1, a_2, a_3, g, q_1, q_2, q_3, q_4, \omega_1, \omega_2, \omega_3, b_1, c_1, b_2, c_2, b_3, c_3, b_4, b_5, b_6, c_6]^{\mathrm{T}},$$

where $[p_1, \cdots, \omega_3]$ comprises from the physical system model Eq. (4.2), $[b_1, c_1]$ comprises the noise states for the x-gyroscope, $[b_2, c_2]$ comprises the noise states for the y-gyroscope, and $[b_3, c_3]$ comprises the noise states for the z-gyroscope, all from Eq. (5.6), $[b_4]$ is the noise state for the x-accelerometer described in Eq. (5.7), $[b_5]$ comprises the noise state for the y-accelerometer described in Eq. (5.8), $[b_6, c_6]$ comprises the noise states for the z-accelerometer described in Eq. (5.9). The x-gyroscope, y-gyroscope, and z-gyroscope measure the rotational velocities ω_1, ω_2, and ω_3 respectively. The x-accelerometer, y-accelerometer, and z-accelerometer measure the acceleration in the directions $\hat{x}_\ell$, $\hat{y}_\ell$, and $\hat{z}_\ell$ respectively.

In total the system has 27 states, 17 for the Newtonian physics portions and 10 for the various noise models. These subsystems are not connected so when we combine them $\mathbf{\Phi}$ has

a block diagonal form.

$$\boldsymbol{\Phi} = \begin{pmatrix} \boldsymbol{\Phi}_{1\cdots17,1\cdots17} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Phi}_{18\cdots19,18\cdots19} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \boldsymbol{\Phi}_{20\cdots21,20\cdots21} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \boldsymbol{\Phi}_{22\cdots23,22\cdots23} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \Phi_{24,24} & 0 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & 0 & \Phi_{25,25} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \boldsymbol{\Phi}_{26\cdots27,26\cdots27} \end{pmatrix},$$

where the submatrices are defined

$$\boldsymbol{\Phi}_{1\cdots17,1\cdots17} = \breve{\boldsymbol{\Phi}} \text{ from Eq. (4.6)} \tag{7.1a}$$

$$\begin{aligned} \boldsymbol{\Phi}_{18\cdots19,18\cdots19} &= \boldsymbol{\Phi}_{20\cdots21,20\cdots21} \\ &= \boldsymbol{\Phi}_{22\cdots23,22\cdots23} \\ &= \breve{\boldsymbol{\Phi}} \text{ seen in Eq. (5.6a)} \end{aligned} \tag{7.1b}$$

$$\Phi_{24,24} = \breve{\boldsymbol{\Phi}} \text{ seen in Eq. (5.7a)} \tag{7.1c}$$

$$\Phi_{25,25} = \breve{\boldsymbol{\Phi}} \text{ see in Eq. (5.8a)} \tag{7.1d}$$

$$\boldsymbol{\Phi}_{26\cdots27,26\cdots27} = \breve{\boldsymbol{\Phi}} \text{ seen in Eq. (5.9a).} \tag{7.1e}$$

The forcing function $\boldsymbol{\Gamma u}$ is completely defined in Eqs. (4.9) and (4.11) as there are no additional components needed from the discussion of the noise states discussed in Chapter 5.

$$\boldsymbol{\Gamma} = \begin{pmatrix} \breve{\boldsymbol{\Gamma}} \\ \mathbf{0} \end{pmatrix} \text{ from Eq. (4.9)} \tag{7.2a}$$

$$\boldsymbol{u} = \breve{\boldsymbol{u}} \text{ from Eq. (4.11)} \tag{7.2b}$$

Just as for the sensor noise states, shown in the Allan Deviation plots, the model for the Newtonian systems 'innovation' noise may be sample rate dependent. We will first define the sample rate and use a simple WGN process to model the innovation.

7.2 Sample Rate

When choosing a sample rate for the filter we must consider a couple of things. Choosing a sample rate which is too small, that is, samples are too infrequent, will result in inaccuracies from the assumptions made during the discretizations of the continuous system in

Chapter 4. On the other hand increasing the sample rate increases the amount of computation we need to do. For our filter we will be using the IMU's sample rate, for example 200Hz in some of the cases we treated. Typically greater than 100Hz will be fast enough to keep the discretization errors small according to the assumptions in Chapter 4.

7.3 System Innovation

With the sample rate defined we are ready to define the system's 'innovation' noise, $\boldsymbol{\Lambda w}$. From Eq. (4.10) and Eq. (4.7) we find definitions of $\breve{\boldsymbol{\Lambda}}\breve{\boldsymbol{w}}$ but we will need to discusses what their physical meaning is and what appropriate values for the covariance matrix $\breve{\boldsymbol{Q}}$ are. Recall from the definition of $\boldsymbol{y}$ from Eq. (4.3) that the physical quantities that we must model are jerk and rotation acceleration, $\boldsymbol{j}$ and $\boldsymbol{\alpha}$ respectively. Based on some knowledge about the types of objects we intend on tracking we can develop the following rudimentary model.

The system's jerk we assume to be a zero mean WGN, with a 1σ value of $6\frac{\mathrm{m}}{\mathrm{s}^3}$. That is to say that approximately 68% of the time we expect the system to be staying within one half the jerk of going from not accelerating to free fall. For rotational acceleration we will again apply a simple WGN model, zero mean with a 1σ value of $\pi\frac{\mathrm{rad}}{\mathrm{s}}$. These numbers are chosen based on observations of both real data and filter performance.

With these models and Eqs. (5.6b), (5.7b), (5.8b), and (5.9b) we are ready define the matrices $\boldsymbol{\Lambda}$ and $\boldsymbol{Q}$.

$$\boldsymbol{\Lambda} = \begin{pmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \\ \mathbf{0} & \boldsymbol{\Lambda}_{7\cdots9,1\cdots3} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \boldsymbol{\Lambda}_{15\cdots17,4\cdots6} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \boldsymbol{\Lambda}_{18\cdots19,7\cdots8} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \boldsymbol{\Lambda}_{20\cdots21,9\cdots10} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \boldsymbol{\Lambda}_{22\cdots23,11\cdots12} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \Lambda_{24,13} & 0 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & 0 & \Lambda_{25,14} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \boldsymbol{\Lambda}_{26\cdots27,15} \end{pmatrix} \tag{7.3}$$

$$\begin{aligned} \boldsymbol{\Lambda}_{7\cdots9,1\cdots3} &= \boldsymbol{\Lambda}_{15\cdots17,4\cdots6} \\ &= \boldsymbol{I} \end{aligned} \tag{7.4a}$$

$$\begin{aligned} \boldsymbol{\Lambda}_{18\cdots19,7\cdots8} &= \boldsymbol{\Lambda}_{20\cdots21,9\cdots10} \\ &= \boldsymbol{\Lambda}_{22\cdots23,11\cdots12} \\ &= \boldsymbol{I} \end{aligned} \tag{7.4b}$$

$$\begin{aligned} \Lambda_{24,13} &= \Lambda_{25,14} \\ &= 1 \end{aligned} \tag{7.4c}$$

$$\boldsymbol{\Lambda}_{26\cdots27,15} = \boldsymbol{I} \tag{7.4d}$$

$$\boldsymbol{Q} = \begin{pmatrix} \boldsymbol{Q}_{1\cdots3,1\cdots3} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{Q}_{4\cdots6,4\cdots6} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \boldsymbol{Q}_{7\cdots8,7\cdots8} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \boldsymbol{Q}_{9\cdots10,9\cdots10} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \boldsymbol{Q}_{11\cdots12,11\cdots12} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & Q_{13,13} & 0 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & 0 & Q_{14,14} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \boldsymbol{Q}_{15\cdots16,15\cdots16} \end{pmatrix} \tag{7.5}$$

$$\boldsymbol{Q}_{1\cdots3,1\cdots3} = 100 \cdot \boldsymbol{I} \tag{7.6a}$$

$$\boldsymbol{Q}_{4\cdots6,4\cdots6} = \pi^2 \cdot \boldsymbol{I} \tag{7.6b}$$

$$\begin{aligned} \boldsymbol{Q}_{7\cdots8,7\cdots8} &= \boldsymbol{Q}_{9\cdots10,9\cdots10} \\ &= \boldsymbol{Q}_{11\cdots12,11\cdots12} \\ &= E(\breve{\boldsymbol{w}}^{\mathrm{T}}\breve{\boldsymbol{w}}) \text{ from Eq. (5.6b)} \end{aligned} \tag{7.6c}$$

$$Q_{13,13} = E(\breve{w}^{\mathrm{T}}\breve{w}) \text{ from Eq. (5.7b)} \tag{7.6d}$$

$$Q_{14,14} = E(\breve{w}^{\mathrm{T}}\breve{w}) \text{ from Eq. (5.8b)} \tag{7.6e}$$

$$\boldsymbol{Q}_{15\cdots16,15\cdots16} = E(\breve{\boldsymbol{w}}^{\mathrm{T}}\breve{\boldsymbol{w}}) \text{ from Eq. (5.9b)} \tag{7.6f}$$

7.4 Observation/Measurement Stacking

At each time index i there is often many different observations/measurements being made so it will be helpful to discuss them each individually and then, later, combine them together. For instance the IMU measures certain quantities at contiguous discrete moments but the idea of a Zero Velocity Update (ZUPT), discussed later in Section 8.2, extends for

an entire time interval, while we may only get infrequent position updates. We can stack two sub measurements at a time i by combining their respective matrices/vectors.

Consider two observations,

$$\boldsymbol{u} = \boldsymbol{\mathcal{A}}\boldsymbol{x} + \breve{\boldsymbol{n}}, E(\breve{\boldsymbol{n}}\breve{\boldsymbol{n}}^{\mathrm{T}}) = \boldsymbol{\mathcal{P}}$$
$$\boldsymbol{v} = \boldsymbol{\mathcal{B}}\boldsymbol{x} + \breve{\boldsymbol{n}}, E(\breve{\boldsymbol{n}}\breve{\boldsymbol{n}}^{\mathrm{T}}) = \boldsymbol{\mathcal{Q}}.$$

These two can be combined into one observation

$$\boldsymbol{z} = [\boldsymbol{u}, \boldsymbol{v}]$$
$$\boldsymbol{H} = \begin{pmatrix} \boldsymbol{\mathcal{A}} \\ \boldsymbol{\mathcal{B}} \end{pmatrix}$$
$$\boldsymbol{R} = \begin{pmatrix} \boldsymbol{\mathcal{P}} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{\mathcal{Q}} \end{pmatrix}.$$

Given more observations this process can be repeated.

When we define an observation it is with this understanding that we will later be able to combine it with others to form a single measurement at each time i. That is we could define gyroscope measurements, accelerometer measurements, and position measurements, with the understanding that at time i we may make make one combined measurement of accelerometers and gyroscopes but not position, and then at time $i+1$ we may combine accelerometers and position but not gyroscopes, without needing to define a separate H matrix for each of these 7 combinations.

7.5 Gyroscope Measurements

The gyroscopes each measure one of the angular velocities directly and its noise states with a simple linear equation. The x-gyroscope, for example, measures ω_1 with the following form

$$\breve{z} = \omega_1 + b_1 + c_1 + \breve{n},$$

which can be represented in the following form,

$$\begin{aligned}
\boldsymbol{H}_{1\cdots14} &= \mathbf{0} \\
H_{15} &= 1 \\
\boldsymbol{H}_{16\cdots17} &= \mathbf{0} \\
\boldsymbol{H}_{18\cdots19} &= [1,1] \text{ seen in Eq. (5.6c)} \\
\boldsymbol{H}_{20\cdots27} &= \mathbf{0}
\end{aligned}$$

$$\boldsymbol{R} = \breve{\boldsymbol{R}} = E\big(\breve{n}^2\big) \text{ from Eq. (5.6d).}$$

We may construct a similar sets of equations for all the gyroscopes and combine them all into one larger gyroscope measurement.

$$\boldsymbol{H} = \begin{pmatrix} \mathbf{0} & \boldsymbol{H}_{15\cdots17,1\cdots3} & \boldsymbol{H}_{18\cdots23,1\cdots3} & \mathbf{0} \end{pmatrix} \tag{7.7}$$

$$\boldsymbol{H}_{15\cdots17,1\cdots3} = \boldsymbol{I} \tag{7.8a}$$

$$\boldsymbol{H}_{18\cdots23,1\cdots3} = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix} \tag{7.8b}$$

$$\boldsymbol{R} = \begin{pmatrix} R_{1,1} & 0 & 0 \\ 0 & R_{2,2} & 0 \\ 0 & 0 & R_{3,3} \end{pmatrix} \tag{7.9}$$

$$\begin{aligned}
R_{1,1} &= R_{2,2} \\
&= R_{3,3} \\
&= E\big(\breve{n}^2\big) \text{ from Eq. (5.6d)}
\end{aligned} \tag{7.10}$$

These two matrices define how we observe local rotational velocity with the gyroscopes.

7.6 Accelerometer Measurements

The three accelerometer's measurements are more complicated as they measure the local acceleration, as opposed to the tracked, global ones, in addition to gravity and their own

noise states. Their observations, ζ_1 for the x-accelerometer, ζ_2 for the y-accelerometer, and ζ_3 for the z-accelerometer from Eq. (1.3a) are as follows,

$$\boldsymbol{\zeta} = \boldsymbol{a}_\ell + \boldsymbol{S} \cdot \begin{pmatrix} 0 \\ 0 \\ g \end{pmatrix} + \breve{\boldsymbol{\mu}},$$

From Eq. (1.2b) we can replace $\boldsymbol{a}_\ell$ with the global, tracked, acceleration.

$$\boldsymbol{\zeta} = \boldsymbol{S} \cdot \boldsymbol{a} + \boldsymbol{S} \cdot \begin{pmatrix} 0 \\ 0 \\ g \end{pmatrix} + \breve{\boldsymbol{\mu}}$$

Recalling that the matrix $\boldsymbol{S}$ is represented by $\boldsymbol{\mathcal{S}}(\boldsymbol{q})^{\mathrm{T}}$, recalling that we track the inverse rotation, from Eq. (3.2)

$$\boldsymbol{\zeta} = \boldsymbol{\mathcal{S}}(\boldsymbol{q})^{\mathrm{T}} \cdot \left(\boldsymbol{a} + \begin{pmatrix} 0 \\ 0 \\ g \end{pmatrix} \right) + \breve{\boldsymbol{\mu}}.$$

We have a model of the noise states for each individual accelerometer from Section 5.2.2 which we will use to replace $\breve{\boldsymbol{\mu}}$.

$$\boldsymbol{\zeta} = \boldsymbol{\mathcal{R}}(\boldsymbol{q}) \cdot \left(\boldsymbol{a} + \begin{pmatrix} 0 \\ 0 \\ g \end{pmatrix} \right) + \begin{pmatrix} b_4 \\ b_5 \\ b_6 + c_6 \end{pmatrix} + \boldsymbol{n} \tag{7.11}$$

This is a nonlinear observation and is linearized through Eq. (4.13) using the current estimates of the states.

$$\boldsymbol{z} = \boldsymbol{\zeta} - \boldsymbol{\mathcal{R}}(\tilde{\boldsymbol{q}}) \cdot \left(\tilde{\boldsymbol{a}} + \begin{pmatrix} 0 \\ 0 \\ \tilde{g} \end{pmatrix} \right) + \begin{pmatrix} \tilde{b}_4 \\ \tilde{b}_5 \\ \tilde{b}_6 + \tilde{c}_6 \end{pmatrix} + \boldsymbol{H}\tilde{\boldsymbol{x}} + \boldsymbol{n}, \tag{7.12}$$

where

$$\boldsymbol{H}^{\mathrm{T}} = \begin{bmatrix}
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
2\,\tilde{x}_{11}{}^2 - 1 + 2\,\tilde{x}_{12}{}^2 & 2\,\tilde{x}_{12}\tilde{x}_{13} - 2\,\tilde{x}_{11}\tilde{x}_{14} & 2\,\tilde{x}_{12}\tilde{x}_{14} + 2\,\tilde{x}_{11}\tilde{x}_{13} \\
2\,\tilde{x}_{12}\tilde{x}_{13} + 2\,\tilde{x}_{11}\tilde{x}_{14} & 2\,\tilde{x}_{11}{}^2 - 1 + 2\,\tilde{x}_{13}{}^2 & 2\,\tilde{x}_{13}\tilde{x}_{14} - 2\,\tilde{x}_{11}\tilde{x}_{12} \\
2\,\tilde{x}_{12}\tilde{x}_{14} - 2\,\tilde{x}_{11}\tilde{x}_{13} & 2\,\tilde{x}_{13}\tilde{x}_{14} + 2\,\tilde{x}_{11}\tilde{x}_{12} & 2\,\tilde{x}_{11}{}^2 - 1 + 2\,\tilde{x}_{14}{}^2 \\
2\,\tilde{x}_{12}\tilde{x}_{14} - 2\,\tilde{x}_{11}\tilde{x}_{13} & 2\,\tilde{x}_{13}\tilde{x}_{14} + 2\,\tilde{x}_{11}\tilde{x}_{12} & 2\,\tilde{x}_{11}{}^2 - 1 + 2\,\tilde{x}_{14}{}^2 \\
4\,\tilde{x}_{11}\tilde{x}_7 + 2\,\tilde{x}_{14}\tilde{x}_8 - 2\,\tilde{x}_{13}\,(\tilde{x}_9 + \tilde{x}_{10}) & -2\,\tilde{x}_{14}\tilde{x}_7 + 4\,\tilde{x}_{11}\tilde{x}_8 + 2\,\tilde{x}_{12}\,(\tilde{x}_9 + \tilde{x}_{10}) & 2\,\tilde{x}_{13}\tilde{x}_7 - 2\,\tilde{x}_{12}\tilde{x}_8 + 4\,\tilde{x}_{11}\,(\tilde{x}_9 + \tilde{x}_{10}) \\
4\,\tilde{x}_{12}\tilde{x}_7 + 2\,\tilde{x}_{13}\tilde{x}_8 + 2\,\tilde{x}_{14}\,(\tilde{x}_9 + \tilde{x}_{10}) & 2\,\tilde{x}_{13}\tilde{x}_7 + 2\,\tilde{x}_{11}\,(\tilde{x}_9 + \tilde{x}_{10}) & 2\,\tilde{x}_{14}\tilde{x}_7 - 2\,\tilde{x}_{11}\tilde{x}_8 \\
2\,\tilde{x}_{12}\tilde{x}_8 - 2\,\tilde{x}_{11}\,(\tilde{x}_9 + \tilde{x}_{10}) & 2\,\tilde{x}_{12}\tilde{x}_7 + 4\,\tilde{x}_{13}\tilde{x}_8 + 2\,\tilde{x}_{14}\,(\tilde{x}_9 + \tilde{x}_{10}) & 2\,\tilde{x}_{11}\tilde{x}_7 + 2\,\tilde{x}_{14}\tilde{x}_8 \\
2\,\tilde{x}_{11}\tilde{x}_8 + 2\,\tilde{x}_{12}\,(\tilde{x}_9 + \tilde{x}_{10}) & -2\,\tilde{x}_{11}\tilde{x}_7 + 2\,\tilde{x}_{13}\,(\tilde{x}_9 + \tilde{x}_{10}) & 2\,\tilde{x}_{12}\tilde{x}_7 + 2\,\tilde{x}_{13}\tilde{x}_8 + 4\,\tilde{x}_{14}\,(\tilde{x}_9 + \tilde{x}_{10}) \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 1 \\
0 & 0 & 1
\end{bmatrix}. \tag{7.13}$$

This observation has the additional WGN, $\boldsymbol{n}$, with a covariance matrix,

$$\boldsymbol{R} = \begin{pmatrix} R_{1,1} & 0 & 0 \\ 0 & R_{2,2} & 0 \\ 0 & 0 & R_{3,3} \end{pmatrix} \tag{7.14}$$

$$R_{1,1} = E\big(\breve{n}^2\big) \text{ from Eq. (5.7d)} \tag{7.15a}$$

$$R_{2,2} = E\big(\breve{n}^2\big) \text{ from Eq. (5.8d)} \tag{7.15b}$$

$$R_{3,3} = E\big(\breve{n}^2\big) \text{ from Eq. (5.9d).} \tag{7.15c}$$

This defines how the accelerometer observes the global accelerations and associated noise states.

7.7 Other Observations

We've seen an example of both a linear observation, in the gyroscopes' measurements, and nonlinear ones, in the accelerometers. Any other observations, usually specific to the test or some devices not discussed, will be handled the same way. Additional instruments may be added with any additional states 'added' in the same way the as the continuous time model had states 'added' for the noise models of the gyroscopes and accelerometers.

7.8 Initial Value

We also need to formulate an initial value for the filter, the pair $\bar{\boldsymbol{x}}[0]$, $\boldsymbol{P}_0$. It is important to get this value close to the correct value in order to get good linearizations going forward. In order to initialze the filter we will be using three pieces of information, an estimate of the units position, a set of samples of IMU data from the initial still period, and an estimate of what we will call the 'yaw' angle. We assume that the initial point is known with an uncorrelated WGN with a 2 cm standard deviation. The unit is initially sitting still for the purpose of calibration so both its velocity and acceleration are zero with associated covariance matrix of $1E{-}4\ \mathrm{m}^2\boldsymbol{I}$. Gravity has been calculated for our location to have an acceleration of approximately $9.8033\frac{\mathrm{m}}{\mathrm{s}^2}$, with a variance of $\left(0.01\frac{m}{s^2}\right)^2$.

To choose initial estimate for the rest of the states we will use the two additional pieces of information, the mean of the samples from the IMU and the approximate direction the unit is pointing in, 'yaw'. The mean of the gyroscope measurements is assigned as all gyroscope error with the variance taking into account the number of samples averaged, we tend to overestimate the number as a ZUPT immediately follows the initialization the filter, has enough time to come to its own asymptote, a typical value would be $8E-10$. To form an estimate of the system's orientation and accelerometer biases we will be borrowing a concept from σ-point methods.

7.8.1 Short Introduction to σ-point Methodology

In order to approximate a variable $\boldsymbol{y}$ that is related to some WGN vector $\boldsymbol{v}$, $\boldsymbol{P} = E\big((\boldsymbol{v}-\bar{\boldsymbol{v}})\,(\boldsymbol{v}-\bar{\boldsymbol{v}})^{\mathrm{T}}\big)$, through a function $\boldsymbol{f}$, $\boldsymbol{y}=\boldsymbol{f}(\boldsymbol{v})$, as WGN we can take two approaches. Throughout the thesis we use a linear approximation for $\boldsymbol{f}$ to update the covariance matrix, so $\bar{\boldsymbol{y}}=\boldsymbol{f}(\bar{\boldsymbol{v}})$ and $E\big((\boldsymbol{y}-\bar{\boldsymbol{y}})\,(\boldsymbol{y}-\bar{\boldsymbol{y}})^{\mathrm{T}}\big)=\boldsymbol{F}^{\mathrm{T}}\boldsymbol{P}\boldsymbol{F}$ where $\boldsymbol{F}=\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{v}}(\bar{\boldsymbol{v}})$. This works well as when

the function $\boldsymbol{f}$ is linear in a region large enough to transform the bulk of the probability density of $\boldsymbol{v}$. In cases where it is not but we still wish to estimate the result as a WGN vector a better approximation can be made by using σ-point methods.

We will define the σ-points to the WGN vector $\boldsymbol{v}$ as,

$$\boldsymbol{\Omega}_{:,1} = \bar{\boldsymbol{v}}$$

$$\boldsymbol{\Omega}_{:,1+i} = \bar{\boldsymbol{v}} + \sqrt{\boldsymbol{P}}_{:,i}$$

$$\boldsymbol{\Omega}_{:,1+k+i} = \bar{\boldsymbol{v}} - \sqrt{\boldsymbol{P}}_{:,i}.$$

Notice that all the information of the WGN vector is contained in these points. Their mean is $\bar{\boldsymbol{v}}$ and we can reconstruct the covariance matrix $\boldsymbol{P}$ with the following formula

$$\boldsymbol{P} = \frac{1}{2}\sum_{i=1}^{2k+1}\left(\Upsilon_{:,i} - \bar{\boldsymbol{v}}\right)\left(\Upsilon_{:,i} - \bar{\boldsymbol{v}}\right)^{\mathrm{T}}$$

With this in mind consider passing these σ-points through the function $\boldsymbol{f}$ and then reconstituting the WGN model with their result. The effects of the nonlinearities of the function have been better approximated by evaluating it at points one standard deviation away from the mean.

$$\bar{\boldsymbol{y}} = \frac{1}{2k+1}\sum_{i=1}^{2k+1}\boldsymbol{f}(\Upsilon_{:,i})$$

$$E(\boldsymbol{y} - \bar{\boldsymbol{y}})\left(\boldsymbol{y} - \bar{\boldsymbol{y}}\right)^{\mathrm{T}} = \frac{1}{2}\sum_{i=1}^{2k+1}\left(\boldsymbol{f}(\boldsymbol{\Omega}_{:,i}) - \bar{\boldsymbol{y}}\right)\left(\boldsymbol{f}(\boldsymbol{\Omega}_{:,i}) - \bar{\boldsymbol{y}}\right)^{\mathrm{T}}$$

7.8.2 Rotation and Bias Estimation

The nonlinear function $\boldsymbol{f}$ from the previous method is realized here as a process. Starting with an initial guess for the accelerometer errors we subtract them from the accelerometer measurements, $\boldsymbol{\zeta}$. The result $\boldsymbol{\zeta} - \breve{\boldsymbol{\mu}}$ is then taken to be completely due to gravity and is normalized to form the estimate, $-\tilde{\hat{z}}_{\ell}$. The estimated y-axis, $\tilde{\hat{y}}_{\ell}$ must lie in the plane perpendicular to this and so is completely characterized by a single angle, which we are calling 'yaw'. With this angle given we have fully specified the local coordinate frame's rotation and thus have an estimate for the current orientation quaternion, $\tilde{\boldsymbol{q}}$. Once the rotation is solved for we can go back and estimate the errors associated with this by subtracting the rotated gravity from the accelerometer measurements,

$$\tilde{\breve{\boldsymbol{b}}} = \boldsymbol{\zeta} - \boldsymbol{\mathcal{S}}(\tilde{\boldsymbol{q}})^{\mathrm{T}}\begin{pmatrix}0\\0\\g\end{pmatrix}$$

We make the initial assumption that the accelerometer biases are 0 with a standard deviation of .2 and that the initial measurement for the yaw has a standard deviation of $\frac{\pi}{8}$, values found in practice. Using these as σ-points for the initial guesses we calculated the estimated mean and covariance for the initial point of the filter.

The process is a little complicated so an example follows.
Consider an initial yaw estimate of 0 and an accelerometer measurement of

$$\boldsymbol{\zeta} = [-0.0234, -0.0938, 9.7656]\,.$$

We combine these into a vector

$$\boldsymbol{v} = [0, -0.0234, -0.0938, 9.7656]$$

and assign it a covariance matrix,

$$\boldsymbol{P} = \begin{pmatrix} \left(\frac{\pi}{8}\right)^2 & 0 & 0 & 0 \\ 0 & (.2)^2 & 0 & 0 \\ 0 & 0 & (.2)^2 & 0 \\ 0 & 0 & 0 & (.2)^2 \end{pmatrix}.$$

This gives us the σ-points

$$\boldsymbol{\Omega} = \begin{pmatrix} 0 & \frac{\pi}{8} & -\frac{\pi}{8} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.2 & -0.2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.2 & -0.2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.2 & -0.2 \end{pmatrix}$$

Taking the 5th point by way of example we will step through the process.
First we subtract the initial estimate of the error from the measurement

$$[-0.0234, -0.0938, 9.765] - [-.2, 0, 0] = [-0.2234, -0.0938, 9.765]\,.$$

We then normalize this to

$$[-0.0229, -0.0096, 0.9997]\,.$$

Assuming that this corresponds to the $-\hat{z}$ direction along with the current estimate for the yaw 0 we find the rotation matrix to be,

$$\boldsymbol{R}^{\mathrm{T}} = \begin{pmatrix} 0.9997 & 0.0000 & -0.0229 \\ -0.0002 & 1.0000 & -0.0096 \\ 0.0229 & 0.0096 & 0.9997 \end{pmatrix}$$

This is turned into a estimate of the quaternion (if this quaternion is more than 1 unit away from the first one we pick the 'negative', equivalent quaternion instead),

$$[\tilde{q}_1, \tilde{q}_2, \tilde{q}_3, \tilde{q}_4] = [0.9999, -0.0048, 0.0114, 0.0001]$$

Finally we subtract the rotation of gravity from the measurements to formulate an estimate of the sensors' errors.

$$\left[\tilde{b}_4, \tilde{b}_5, \tilde{b}_6\right] = [0.2008, 0.0003, -0.0346]$$

We repeat the process for all the original σ-points and construct the mean and covariance between the current rotation estimate and accelerometer error.

7.8.3 Error Estimates to Noise State Estimates

The gyroscope and accelerometer noise states are assumed to start at 0 and have had infinite time to settle to their current value. This makes all the bias walks zero mean WGN with infinite variance. The colored noise states are also zero mean WGN noise but have finite variance, $c_1 c_2^2$, found by letting $\epsilon = 0$ in Eq. (5.5c). The estimate of error is treated as a pseudo measurement of the sum of these quantities and some additional WGN; the infinite variance of the bias walk results in all the error being assigned to it.

Chapter 8

Small Scale Test and Zero Velocity Updates

In order to demonstrate the performance of the filter we performed a small scale test, capturing the data of a relatively simple set of motions.

8.1 Test Plan

The test was set up in two distinct parts. First is a sequence of motions designed to aid the filter in estimating the various noise effects and initial states. This is an attempt to reduce the covariance of various terms from the initialization as described in Section 7.8. Immediately after these movements we begin a sequence that is designed to test the filter's ability to track motions.

For the test set up a desk was divided up into a grid of points with the addition of a box at one point to provide height diversity. The desk with an overlay of the labeled points is shown in Figure 8.1. The points' measured coordinate values can be seen in Table 8.1. All the motions during the test are referenced to this grid. At the beginning and end of each motion the unit is placed at one of these points. Each motion/step, therefore, is characterized by the point at which it ends (the start point is implied from the previous motion), the unit's orientation it ends at, and how long the unit was left motionless before the next movement was started. All the movements between points were performed as to take as direct a path as possible with the notable exception of Step 20 in Table 8.3 wherein the laptop used for data recording blocked the direct path and the unit had to travel around

the screen. The unit was picked up and placed by hand and approximate timing was kept through the use of a metronome.

Point	X (m)	Y (m)	Z (m)
1	0.70	0.20	0.00
2	0.40	0.20	0.00
3	0.10	0.20	0.00
4	0.70	0.80	0.00
5	0.40	0.80	0.10
6	0.10	0.80	0.00
7	0.70	1.40	0.00
8	0.40	1.40	0.00
9	0.10	1.40	0.00

Table 8.1: Truth Points

8.1.1 System Calibration

The first part of the test is the sequence of movements designed to aid the system in calibrating itself. These motions are enumerated in Table 8.2. Steps 1-4 are designed with the intent of isolating accelerometer noise from gravity. Steps 5-8 are intended to aid in the determination of the initial 'yaw' orientation, the direction the unit is facing. During each of these steps the filter is provided with Zero Velocity Update (ZUPT) information when the unit is not moving, and position observations as soon as the unit is set at rest.

8.1.2 Tracked Movements

Immediately after the calibration steps the system is carried through a sequence of movements we wish to track. The path of movements can be seen in Figure 8.2 and each step is enumerated in Table 8.3. During the 5 seconds of non-motion at each point the filter is improved with Zero Velocity Update information but no position observations are provided. In total the unit travels over 6.18 m and has a 0 m displacement from its original point.

Step	Point	Face	Time (apprx)
1	1	1	30s
2	1	2	30s
3	1	3	30s
4	1	1	30s
5	2	1	5s
6	1	1	5s
7	4	1	5s
8	1	1	5s

Table 8.2: Calibration Sequence

Step	Point	Face	Time (apprx)
9	2	1	5s
10	3	1	5s
11	6	1	5s
12	9	1	5s
13	8	1	5s
14	5	1	5s
15	7	1	5s
16	4	1	5s
17	5	1	5s
18	7	1	5s
19	4	1	5s
20	1	1	5s

Table 8.3: Tracked Sequence

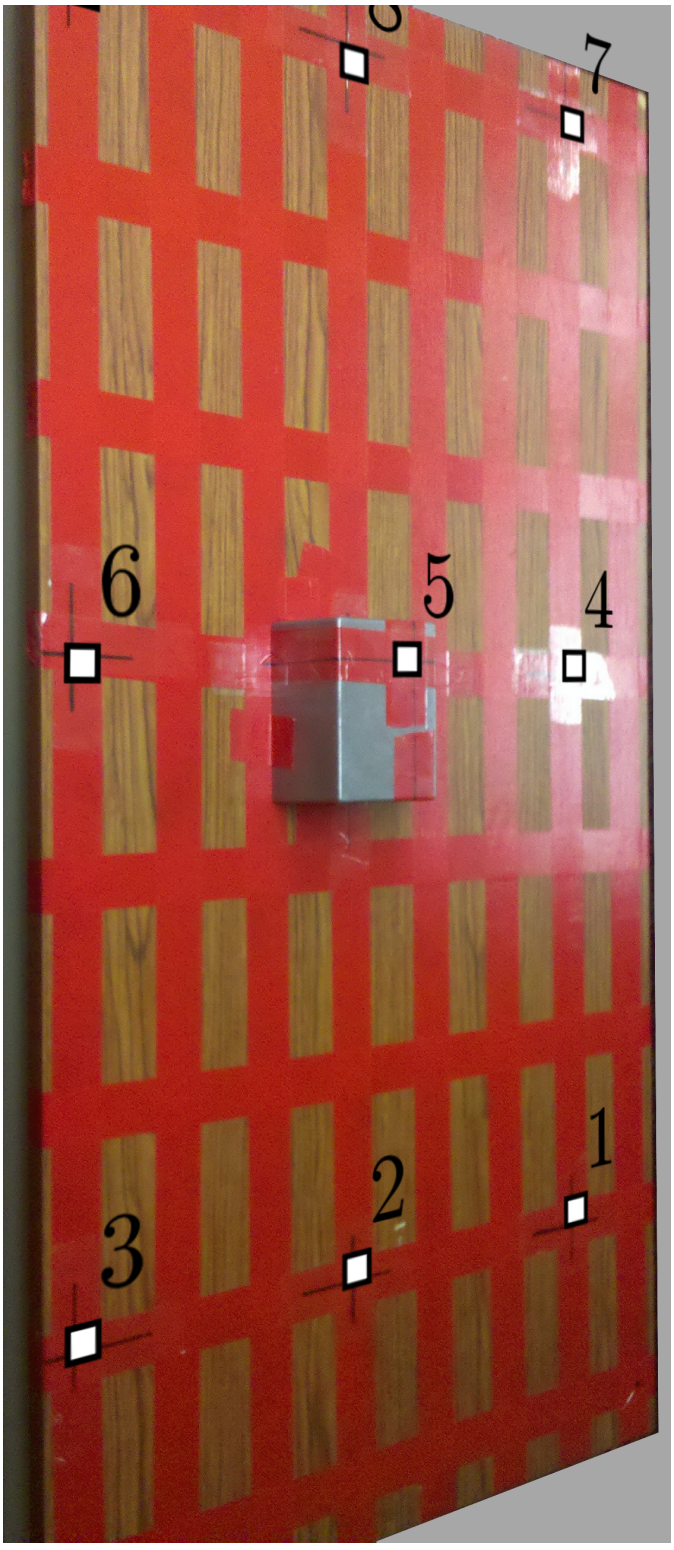

Figure 8.1: Table Layout with Point Labels Overlayed

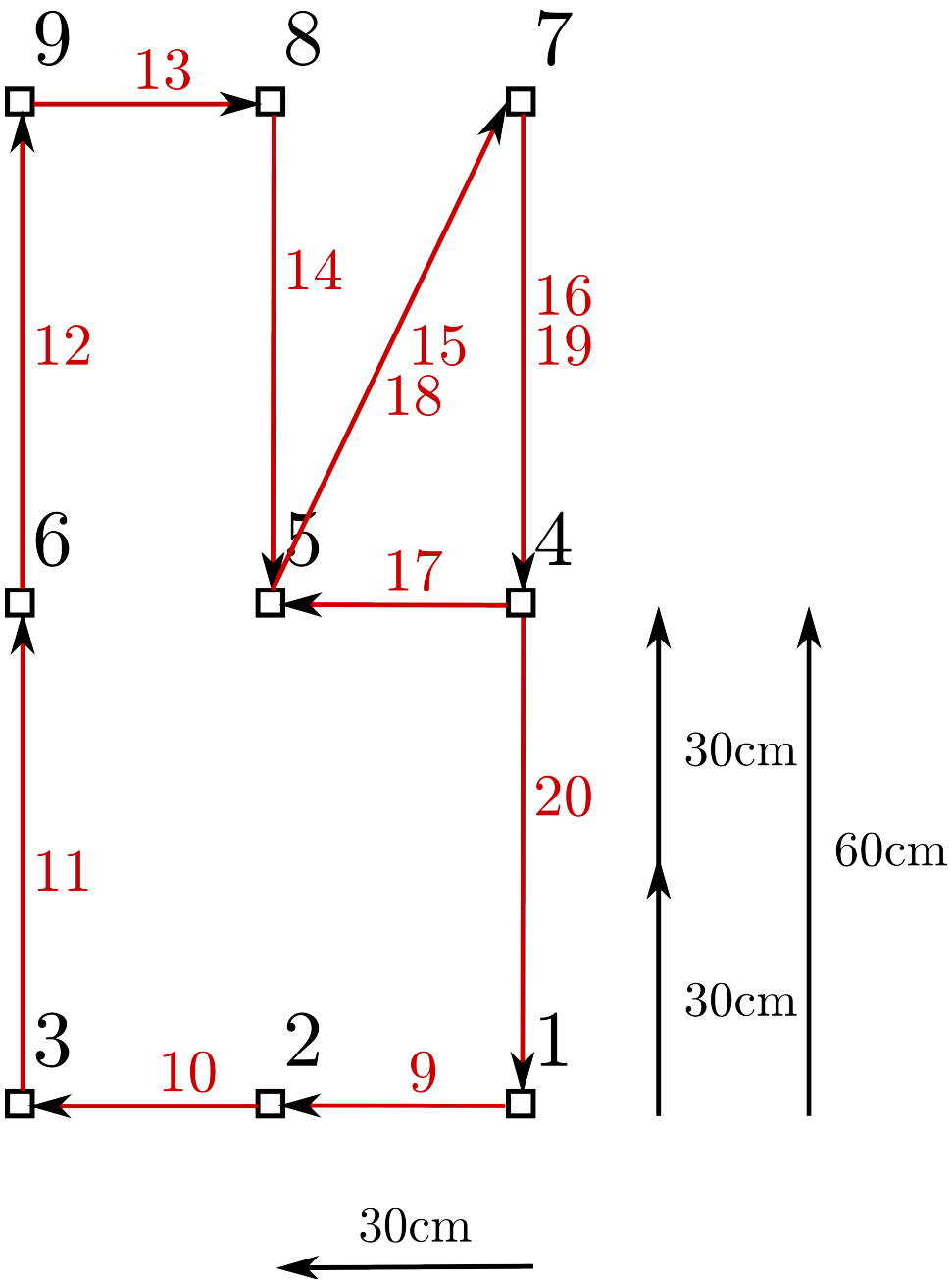

Figure 8.2: Tracked Movements

8.2 Zero Velocity Updates

In order to aid the filters performance we will be providing it with information about the periods of non-motion. This information is referred to as a Zero Velocity Update (ZUPT). We have determined these periods of time by observing the original data collected by the sensor and judging when the periods of rest occurred. An example of a ZUPTing period can be seen in Figure 8.3.

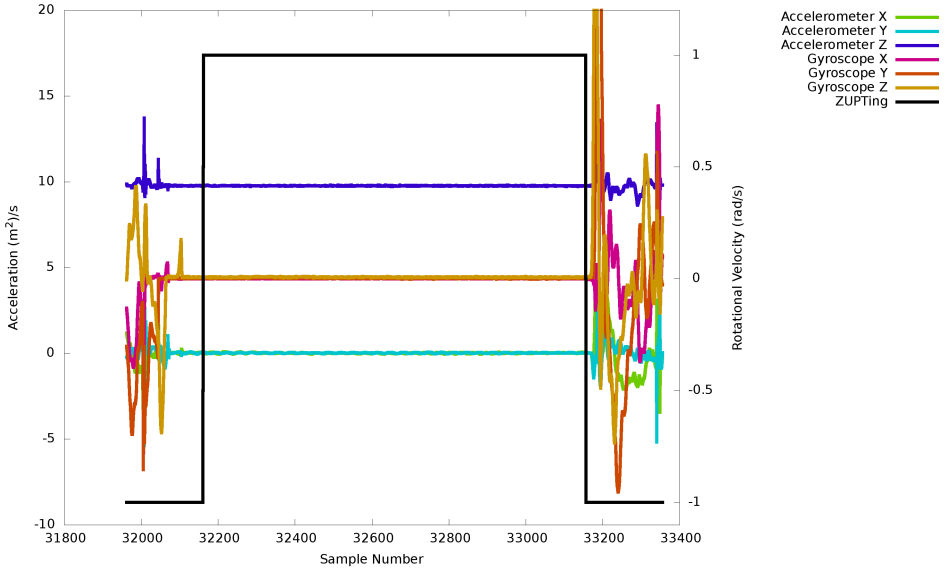

Figure 8.3: Example ZUPT

During periods of ZUPTing the actual motion of the device is much more complicated than what we can model simply and to mitigate this we have broken it into two parts. The first part is a set of observations indicating that the system is not moving; we expect the velocities, accelerations, and rotational velocities to be 0 with small variances. This is only true, however, on the timescale of an entire ZUPT. In the small intervals of time that we are sampling at we cannot expect the device to have such small motions. In Figure 8.4 we can easily see a decaying sinusoidal response associated with a spring-mass system with friction, which in this case is probably due to the swaying of the table on its legs. This second set of observations will be made in lieu of the inertial observations discussed in Sections 7.5 and 7.6, these are shown in Eq. (8.4) and (8.5).

During each time increment during these intervals the filter is provided with a series of

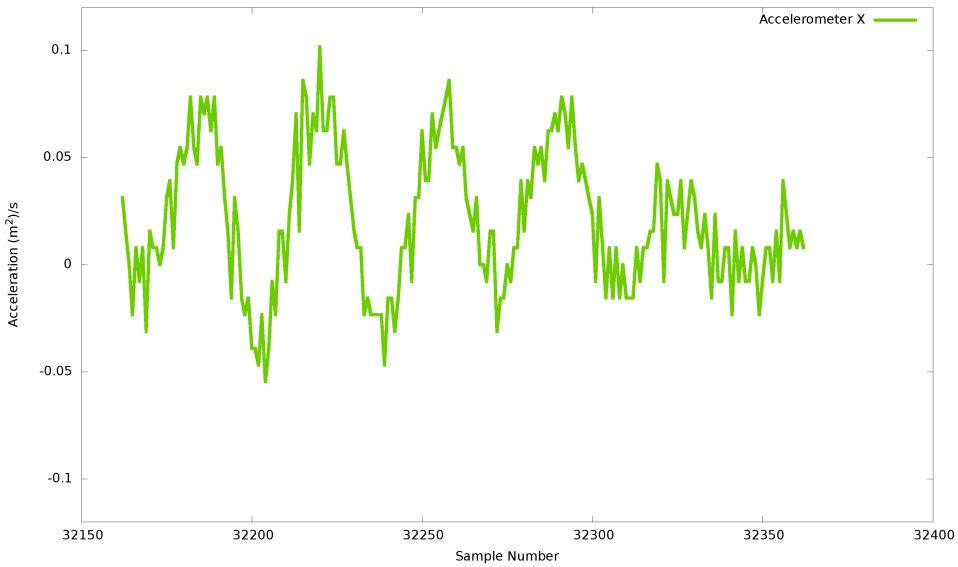

Figure 8.4: Small Movements during ZUPT

'no motion' observations. First the velocity of the system is observed to be $\mathbf{0}$,

$$\boldsymbol{z} = [0, 0, 0] \tag{8.1a}$$

$$\boldsymbol{H} = \begin{pmatrix} \mathbf{0} & \boldsymbol{H}_{4\cdots6,1\cdots3} & \mathbf{0} \end{pmatrix} \tag{8.1b}$$

$$\boldsymbol{H}_{4\cdots6,1\cdots3} = \boldsymbol{I}$$

$$\boldsymbol{R} = (0.0001)\boldsymbol{I}. \tag{8.1c}$$

Additionally the system is observed to be not accelerating,

$$\boldsymbol{z} = [0, 0, 0] \tag{8.2a}$$

$$\boldsymbol{H} = \begin{pmatrix} \mathbf{0} & \boldsymbol{H}_{7\cdots9,1\cdots3} & \mathbf{0} \end{pmatrix} \tag{8.2b}$$

$$\boldsymbol{H}_{7\cdots9,1\cdots3} = \boldsymbol{I}$$

$$\boldsymbol{R} = (0.00025)\boldsymbol{I}. \tag{8.2c}$$

The system is also not rotating,

$$\boldsymbol{z} = [0, 0, 0] \tag{8.3a}$$

$$\boldsymbol{H} = \begin{pmatrix} \mathbf{0} & \boldsymbol{H}_{15\cdots17,1\cdots3} & \mathbf{0} \end{pmatrix} \tag{8.3b}$$

$$\boldsymbol{H}_{15\cdots17,1\cdots3} = \boldsymbol{I}$$

$$\boldsymbol{R} = (1E-6)\boldsymbol{I}. \tag{8.3c}$$

The motions of the accelerometers and gyroscopes actually experienced is modeled as WGN with a larger variance, to take into account the actual small time scale movements.

$$\begin{aligned} \boldsymbol{\zeta} &= \text{Acceleromter Measurements} \\ \boldsymbol{\zeta} &= \boldsymbol{\mathcal{R}}(\boldsymbol{q}) \cdot \begin{pmatrix} 0 \\ 0 \\ g \end{pmatrix} + \begin{pmatrix} b_4 \\ b_5 \\ b_6 + c_6 \end{pmatrix} + \boldsymbol{n} \end{aligned} \tag{8.4a}$$

$$\boldsymbol{R} = (0.01)\boldsymbol{I}, \tag{8.4b}$$

and

$$\boldsymbol{z} = \text{Gyroscope Measurements} \tag{8.5a}$$

$$\boldsymbol{H} = \begin{pmatrix} \mathbf{0} & \boldsymbol{H}_{18\cdots23,1\cdots3} & \mathbf{0} \end{pmatrix}$$

$$\boldsymbol{H}_{18\cdots23,1\cdots3} = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix} \tag{8.5b}$$

$$\boldsymbol{R} = (0.0025)\boldsymbol{I}. \tag{8.5c}$$

The idea here is similar to that encountered with respect to the systems described by an Allan Variance; on the timescale of $2-5$ seconds the variance for the ZUPT is different from the variance seen on the $\frac{1}{100} - \frac{1}{200}$ second scale.

8.2.1 Additional Adjustments

In the processing of the data there were some additional steps needed. During the Steps 1-4 as the sensor was rotated but not actually moved, the system estimated 'phantom' movements, see Figure 8.5 for an example of such a movement during Step 4. These are caused by incorrect initial estimates of states. From the filter's perspective however the movements are real and it makes some incorrect judgments from them. These movements,

if real, when followed by a position estimate would allow one to estimate the 'yaw' angle. At the end of the movement the filter does receive estimates of its current position and attempts to uses them to estimate its 'yaw'. The system has not actually moved so this measurement is actually impossible which causes the filters attempt to be incorrect. To compound things this also provides the filter with a phantom confidence in its 'yaw' estimate, limiting a future measurement's impact. To fix this at the end of Step 4 we add a 'burst' of variance to the covariance matrix $\boldsymbol{P}$. We take the current rotation quaternion estimate, perturb it by $\pm\frac{\pi}{4}$ radians, a value effectively found through experimentation, and construct a covariance matrix for these perturbations as discussed in Section 7.8. This covariance is added directly to the current covariance matrix as a way to reassert the uncertainty in the 'yaw' that has been erroneously removed by the 'phantom' movements. Additionally a measurement is made of the current rotation quaternion with the initially estimated 'yaw', 0, to prevent further linearization errors.

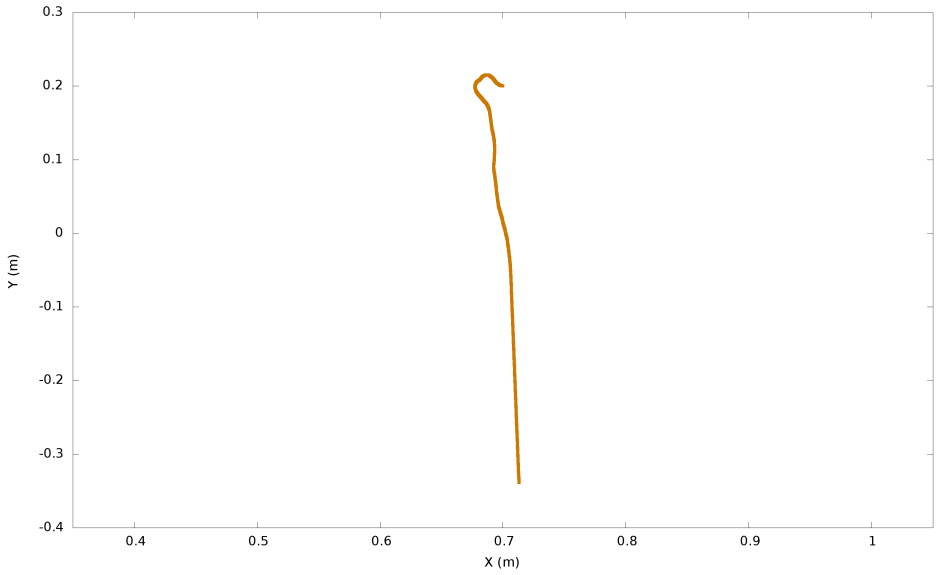

Figure 8.5: Phantom Movement in Step 4

8.3 Percent Error per unit Distance Traveled

As a metric for evaluating how well the filter performs we will be using the percent error per unit distance traveled. This is often used as the figure of merit for tracking systems in

the field[9].

8.4 Filter Performance

The filter designed in this thesis has its performance evaluated in Table 8.4. The filter's estimated x, y, and z values can be seen verses time in Figures 8.6, 8.7, 8.8 respectively. Additionally x vs. y track can be seen 8.9. The error at the last point was $[2, 3, 10]$ cm which is encouraging because the 1σ value tracked by the filter for both the x and y estimates, 2.09 cm and 2.07 cm respectively, indicate that the final point was within 2 standard deviations of our estimate. This will be important in the future as other systems are integrated into the filter so the current estimates will be properly weighted against their measurements. The larger error in the z direction is expected because of the large gravitation acceleration in this direction, its tracked 1σ value, however, appears to be unusually small at 1.26 cm, indicating our error was more than 7σ.

Step	Distance Traveled	True Position	Estimated Position	Error	% Error
9	0.30	$[0.40, 0.20, 0.00]$	$[0.40, 0.20, 0.01]$	0.01	3.14
10	0.60	$[0.10, 0.20, 0.00]$	$[0.10, 0.20, 0.01]$	0.01	2.17
11	1.20	$[0.10, 0.80, 0.00]$	$[0.09, 0.80, 0.03]$	0.04	2.92
12	1.80	$[0.10, 1.40, 0.00]$	$[0.09, 1.41, 0.05]$	0.05	2.56
13	2.10	$[0.40, 1.40, 0.00]$	$[0.40, 1.40, 0.03]$	0.03	1.45
14	2.70	$[0.40, 0.80, 0.10]$	$[0.41, 0.80, 0.13]$	0.03	1.21
15	3.37	$[0.70, 1.40, 0.00]$	$[0.74, 1.40, 0.04]$	0.06	1.67
16	3.39	$[0.70, 0.80, 0.00]$	$[0.72, 0.80, 0.05]$	0.05	1.38
17	4.27	$[0.40, 0.80, 0.10]$	$[0.43, 0.81, 0.15]$	0.06	1.47
18	4.94	$[0.70, 1.40, 0.00]$	$[0.74, 1.41, 0.06]$	0.07	1.45
19	5.54	$[0.70, 0.80, 0.00]$	$[0.74, 0.81, 0.06]$	0.07	1.29
20	6.14	$[0.70, 0.20, 0.00]$	$[0.68, 0.23, 0.10]$	0.10	1.63

Table 8.4: New Filter's Track Estimates

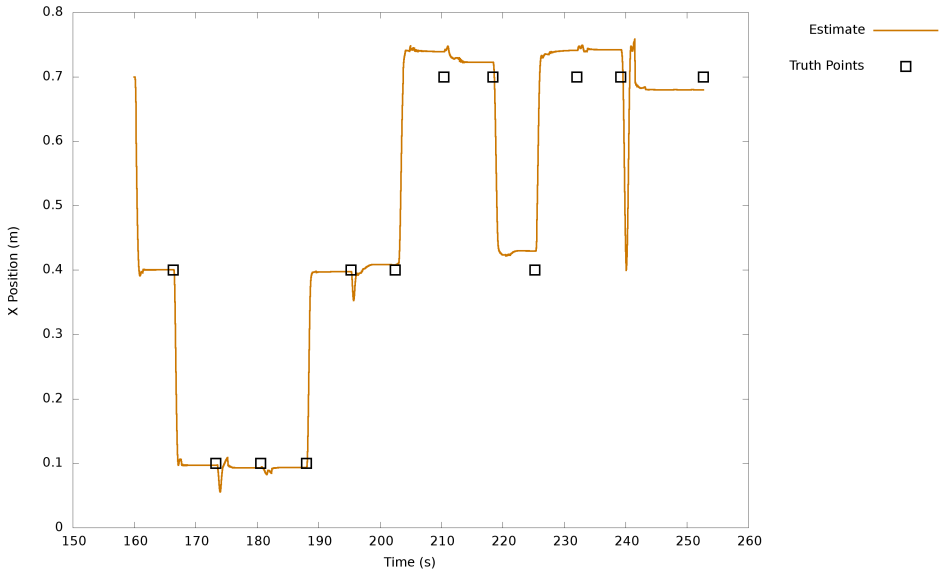

Figure 8.6: New Filter's Estimated X

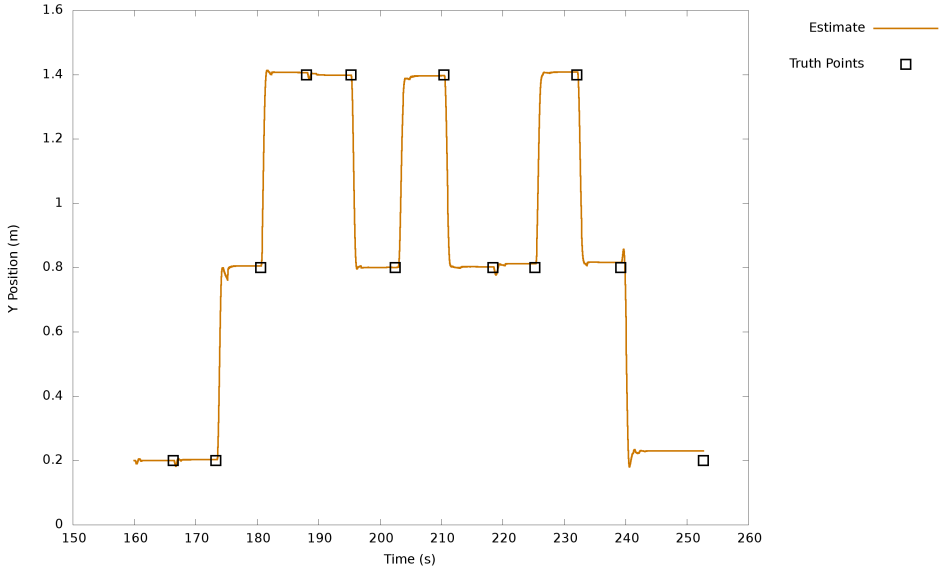

Figure 8.7: New Filter's Estimated Y

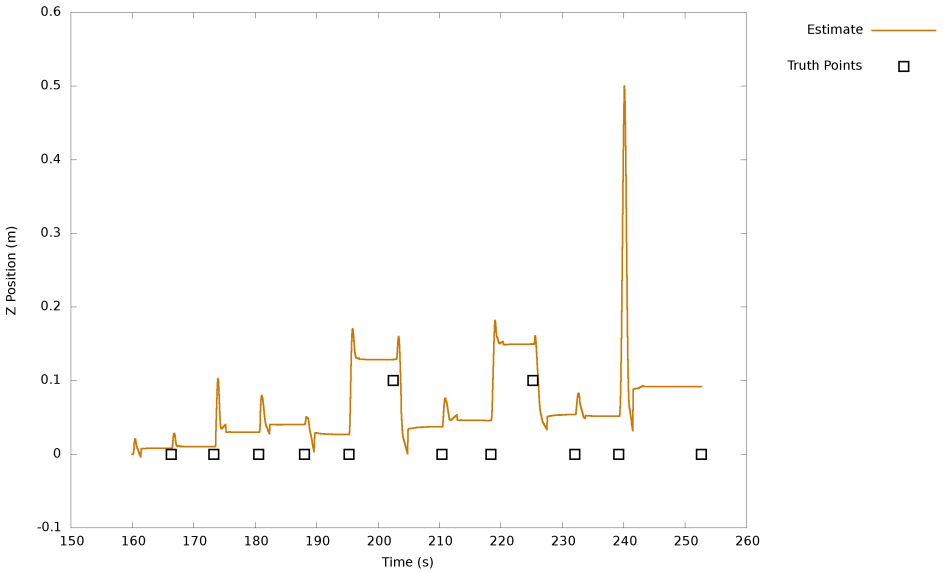

Figure 8.8: New Filter's Estimated Z

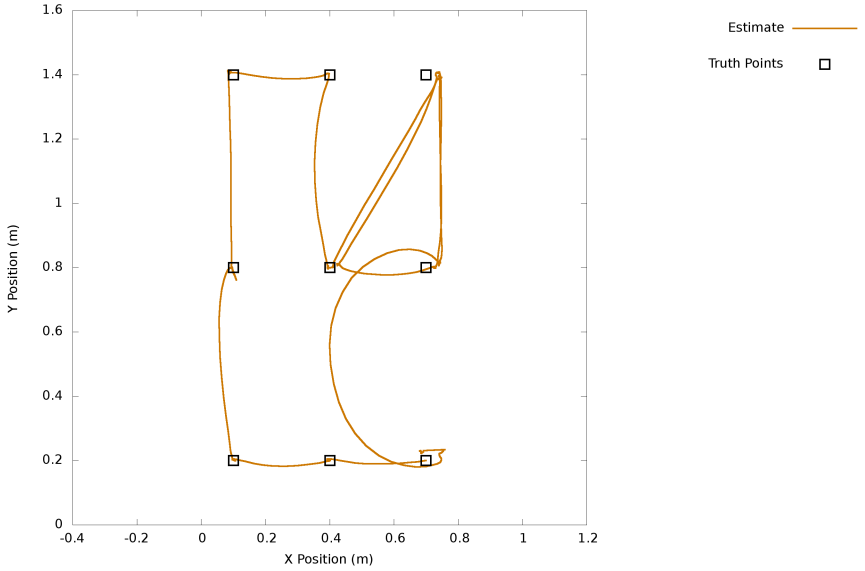

Figure 8.9: New Filter's Estimated XY

Chapter 9

Maple™ Implementation of Discussed Methods

At this point it should be noticed that the work of the previous chapters can be represented as a set of 4 'variables' and a process we apply to those variables. The 'variables' are *sample rate*, the *continuous time system*, the *noise specifications*, and the *observation equations*. Once these variables are defined we can generate our filter by performing the same series of steps; linearization, discretization, model building, and combination. As we look forward to implementing systems using more and different sensor packages it will be beneficial to automate the formulaic manipulations of these variables. This serves two purposes, first it greatly reduces the amount of work needed to realize an implementation and second it reduces the chance of an error being suffered in one of the necessary steps.

The formulas in this chapter should be taken as mimicking Maple™ code and will not have typical mathematical notation. When Maple™ code is listed it will be listed in the following format.

```
f := (x) -> x^2;
g := (x) -> x^2; #Comment
```

9.1 Maple™

Maple™ is a computer algebra system distributed by Maplesoft™. This software was chosen because of its inbuilt functions, including a linear algebra library, simultaneous equation solver, symbolic differentiation/integration, and its functional programming inspired

scripting language. It also is capable of generating Matlab code which can be used directly in data processing.

9.2 Variables

The first variable related to the system we will treat is the *sample rate* of the system, as this will impact our models for the *continuous time system* and *noise specifications*. As previously discussed, the sample rate needs to be fast enough that the approximations that we make will be valid. The sample rate is defined as its inverse, sample period, `epsilon=1/117` for example in a system with a sample rate of 117 Hz, the sample rate used in our implementation of the ADIS16375, the device used in this example.

With the sample period defined we can move on to defining the *continuous time system*. It will be an list of array variables, `c_model = [x,Dx,y,Q]`, where `x` is the state of the system, `Dx` is its derivative, `y` is the forcing function, and `Q` is the covariance matrix of the forcing function, `y`, on the sample period `epsilon`. For example,

```
x := [p[1],p[2],p[3],
v[1],v[2],v[3],
a[1],a[2],a[3],
g,
q[1],q[2],q[3],q[4],
ot[1],ot[2],ot[3]];

Dx := map((var) -> D || var,x); #Build the list [Dp[1],Dp[2], ... ect.
Dp[1] := v[1]:
Dp[2] := v[2]:
Dp[3] := v[3]:
Dv[1] := a[1]:
Dv[2] := a[2]:
Dv[3] := a[3]:
Da[1] := j[1]:
Da[2] := j[2]:
Da[3] := j[3]:
Dg := 0:
#Define the rotational quaternion derivative matrix
Omega := -1/2*Matrix([[0,ot[1],ot[2],ot[3]],
```

```
[-ot[1],0,-ot[3],ot[2]],
[-ot[2],ot[3],0,-ot[1]],
[-ot[3],-ot[2],ot[1],0]]):
Dq[1] := Omega[1,1]*q[1] + Omega[1,2]*q[2] + Omega[1,3]*q[3] + Omega[1,4]*q[4]:
Dq[2] := Omega[2,1]*q[1] + Omega[2,2]*q[2] + Omega[2,3]*q[3] + Omega[2,4]*q[4]:
Dq[3] := Omega[3,1]*q[1] + Omega[3,2]*q[2] + Omega[3,3]*q[3] + Omega[3,4]*q[4]:
Dq[4] := Omega[4,1]*q[1] + Omega[4,2]*q[2] + Omega[4,3]*q[3] + Omega[4,4]*q[4]:
Dot[1] := tt[1]:
Dot[2] := tt[2]:
Dot[3] := tt[3]:

y := [j[1],j[2],j[3],tt[1],tt[2],tt[3]];

Q := DiagonalMatrix([50^2,50^2,50^2,(5*Pi)^2,(5*Pi)^2,(5*Pi)^2]);

c_model := [x,Dx,y,Q];
```

This represents our continuous time model and could be expanded easily to include any additional states which can be defined in this way or could be changed to use a different rotation charting.

We can now define the *noise specifications* for the various noise sources. The final variable `noise` will be a list of these noise sources. Each noise variable will be defined as a `name,[specification],sample_period`. The `name` is the variable name for the noise sources, so for instance `gyn[1]` for the first gyroscope's noise. The list `[specification]` can be one of two different things, either the list of Allan deviation 1 second crossing points and the number of colored noise variables to use to model the flicker region `[wgn,[flicker,#c],bias_walk]` or as simply the WGN variance `[wgn]`. The last, the `sample_period`, defines the sample period of the sensor. Recall that it is from the sensor sample rate that the WGN variance is determined not the system's (use a value of 1 when a simple WGN specification is used). For example,

```
#All the gyros have the same basic gyro noise specification (using 1 colored noise)
bgn := [2.9E-4,[4.8E-5,1],1E-6]:
#Same with the accelerometers (using 2 colored noise)
ban := [4.9E-4,[10E-4,2],2E-5]:
#But each gyro and accelerometer has its own noise variable
```

```
#Sample rate of the filter is the same as the IMU
gyro_noise := [gyn[1],bgn,epsilon],[gyn[2],bgn,epsilon],[gyn[3],bgn,epsilon];
accel_noise := [an[4],ban,epsilon],[an[5],ban,epsilon],[an[6],ban,epsilon];

#Some additional noise variables, such as the noise on a good position update or ZUPT
good_position := [gm[1],[0.01^2],1],[gm[2],[0.01^2],1],[gm[3],[0.01^2],1];
zupt_velocity := [zv[1],[0.01^2],1],[zv[2],[0.01^2],1],[zv[3],[0.01^2],1];

#Leaving our noise variable
noise := [gyro_noise,accel_noise,good_position,zupt_velocity]:
```

The last variable we will be defining is that containing the *observation equations*. This definition will be broken up into two parts, linear observation equations and nonlinear observation equations. Both variables will be defined in the same way, `["Matlab_Name",[m1,m2,...]]`, where `"Matlab_Name"` is the function in Matlab we will be able to call with our observation and `m1,m2,...` are the observation equations. For example,

```
#gyros measure rotation velocity, ot, and their noise, gyn
#zupt is a measurement of velocity, v, and some small velocity noise, zv
#good position measurement the position, p, and some small measurement noise, gm
l_observations := [["Gyro_Measurement",[ot[1]+gyn[1],ot[2]+gyn[2],ot[3]+gyn[3]]],
["Zupt_Velocity",[v[1]+zv[1],v[2]+zv[2],v[3]+zv[3]]]
["Good_Position",[p[1]+gm[1],p[2]+gm[2],p[3]+gm[3]]]]:

#Make a rotation matrix from our quaternion
Rot := quat2rotm([q[1],q[2],q[3],q[4]]);
#The nonlinear observation of accelration R(a+g)+an
nl_observations := [["Accel_Measurement",convert(
Transpose(Rot).Matrix([[a[1]],[a[2]],[a[3]+g]])+Matrix([[an[4]],[an[5]],[an[6]]])
,'list')]]:
```

Notice that in the example we defined our continuous time system's variable names and our noise specification names to be used in later processing.

9.2.1 Filter Creation

The full process of filter creation is described here with a basic flow chart shown in Fig. 9.1.

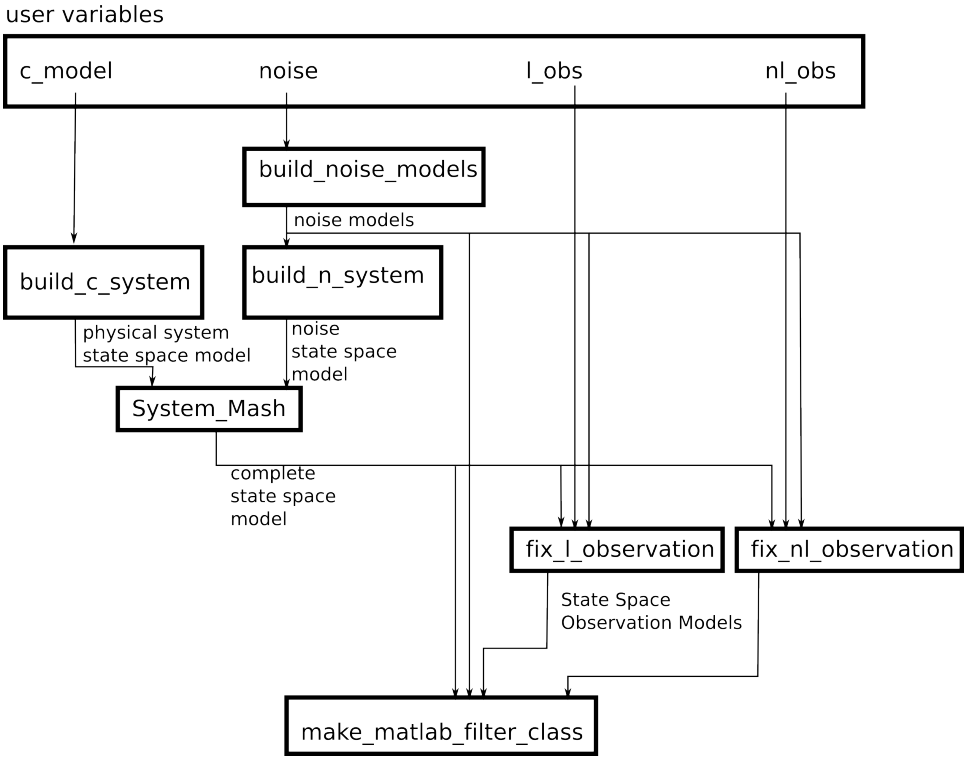

Figure 9.1: Maple Script Flow Chart

The first step is the creation of the noise models from the noise specifications.

```
noise_models := build_noise_models(noise,epsilon);
```

The function `build_noise_models` creates the necessary noise models from the specified values as we did in Chapter 5. It creates a list of all the WGN, colored noise, and bias walks needed to model the various noise specifications along with the equivalency that relates each 'specified noise', such at `gyn[1]`, with its associated parts, `gyn[1]=gn[1]+cn[1]+bw[1]`.

These parts of the models which need to have state representations are made into their appropriate state space models with the function `build_n_system`

```
n_system := build_n_system(noise_models);
```

This creates a state space system, `x,Phi,Gamma,u,Lambda,Q,y`.

The continuous time model `c_model` can also be made into a state space model with the function `build_c_system`

```
c_system := build_c_system(c_model,epsilon,n);
```

where `n` is the order of the Taylor series approximation in time of `Phi=exp(F*epsilon)` and `Xi`. The output of this function is also a state space system, `x,Phi,Gamma,u,Lambda,Q,y`.

The two systems, `n_system` and `c_system`, are combined into one complete system, `s_system`, with the function `System_Mash`.

With the complete system and noise models defined we can 'fix' the observation models to be of the complete systems states, formulating the matrices `H` and `R`.

```
fixed_l_obs := fix_l_observation(l_observations,s_system,noise_models);
fixed_nl_obs := fix_nl_observation(nl_observations,s_system,noise_models);
```

We define one additional variable `ostates` which contains the states we measure as part of our initialization at the beginning of a test. These should include all the states of the `c_model` and any noise variables that were specified from Allan Deviation. The starting covariance for the noise model components are computed as they were in Section 7.8.3. Additionally we will need to know what parts of the forcing function `y` have known, non-zero, mean, which for our case is none so `ybar=[]`. With all of this done we can call the function which creates the Matlab class file that we can use to process data with the function `make_matlab_filter_class`.

```
make_matlab_filter_class(Filter_Name,s_system,noise_models,fixed_l_obs,fixed_nl_obs,ybar,osta
```

9.3 Matlab Class

The resulting Matlab Class extends a generic Kalman Filter class created for this project. It is simply instantiated and initialized with the name specified in the `make_matlab_filter_class`, `filter = Xsens_Filter()`, for example. The filter is indexed forward with the function `filter.next_step()` and we can make observations by calling the observation function handles by the names we specified in the Maple™ script, `filter.Zupt_Velocity([0,0,0]')`. After all the observations are made, the function `filter.make_observation()` can be called to perform the correction step. We can simply repeat this process for the duration of the test.

In this way we can create filters for any IMU/system we need, allowing us to test the filter creation method on a new pair of units in the following chapter.

Chapter 10

Pedestrian Motion Tests

In order to demonstrate our filter generation technique we acquired a reference data set made publically available by the German Aerospace Center [4]. To process this data set we formulated our own ZUPT detection algorithm, and with this detection algorithm and our generated filter we were able to achieve results that we could compare against similar published results and the 'truth' data. Incorporating the lessons learned from this data set we moved on to create our own foot mounted inertial navigation solution and performed our own in lab test.

10.1 Reference Data

For this evaluation we used a data set collected by the German Aerospace Center [4]. In the tests a subject wearing an instrumented shoe traversed a test room using several paths. The worn shoe was equipped with an IMU sensor, Xsens MTx-28A53G25, and a set of infrared reflectors, see Fig. 10.1. The room contained an optical tracking system which was used to track the shoe's progress. The optical tracking system's error is assumed to be small enough so as to allow the data to be used as a truth data set against which different algorithms can be compared.

Of the data sets captured we will be focusing on the first two, ID1 "Walk 2D - rectangle" and ID2 "Walk 2D - rectangle other direction". The first, ID1, has an incomplete inertial capture and will be used to to develop the system's ZUPT detection algorithm. The second, ID2, will be processed and our results will be compared with the truth data.

Figure 10.1: German Aerospace Center Instrumented Shoe, from DLR

10.2 Test Configuration

Both data sets are collected with a subject walking similar paths in a rectangular path through the room. In ID1 the subject walked the path in the counterclockwise direction and in ID2, clockwise, see Fig. 10.5 for the path of ID2. Before each test the subject walks to the start point and stamps his/her foot to provide a synchronization event between the optical tracking system's data and the IMU's. There are brief segments where the subject leaves the optical system's field of vision and the shoe becomes untracked. They are infrequent and short enough to not affect the results at all.

10.3 Xsens IMU

The IMU chosen by German Aerospace Center was the Xsens MTx-28A53G25. The specification sheet does not include an Allan Deviation plot and where values are listed which can be used to compute values of interest they are specified for the entire temperature range and are an order of magnitude larger than expected. Though we do not have access to our own unit to perform our own noise analysis we were able to find a resource which did [18]. From their Allan Deviation plots shown in Figs. 10.2 and 10.3 we were able to derive the necessary values for a system model design, shown in Table 10.1. The noise models were derived using the same process as was described in Chapter 5.

	Gaussian Noise	Flicker Noise	Bias Walk
Gyroscopes	$1.4E-3\frac{rad}{s}$	$1.7E-4\frac{rad}{s}$	$1.7E-6\frac{rad}{s}$
Accelerometers	$2E-3\frac{m}{s^2}$	$4E-4\frac{m}{s^2}$	$4E-6\frac{m}{s^2}$

Table 10.1: Table of Allan Deviation 1s Values for Xsens MTx-28A53G25

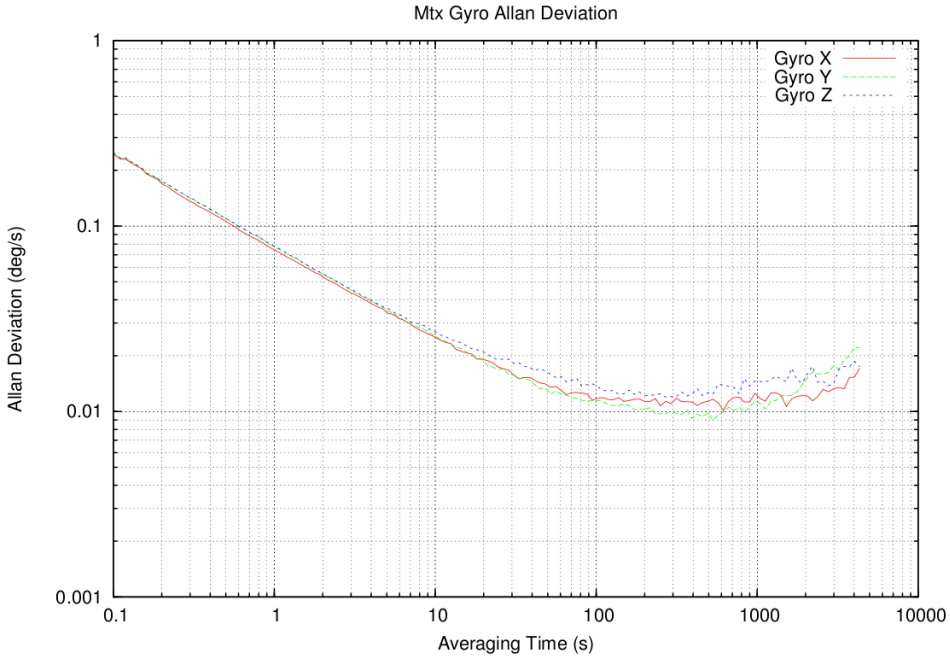

Figure 10.2: Xsens MTx Gyroscope Allan Deviation from woodman

10.4 ZUPT Detection

To fairly compare our results against the similar results in the report we implemented our own automatic ZUPT detection algorithm. The underlying theoretical idea behind our detector is that the sensor is moved by a noise process with non-zero variance, so the variance of the measurements should be larger when the sensor is moving. We use the sample variance of a small window of data around the point of interest and compare it against a threshold. If the variance is under the threshold we assume that the only noise process at work is that of the sensor. If the variance is above the threshold we assume that the sensor is being acted upon by another noise process, motion.

To find good window size and threshold values we used the first data set ID1. We

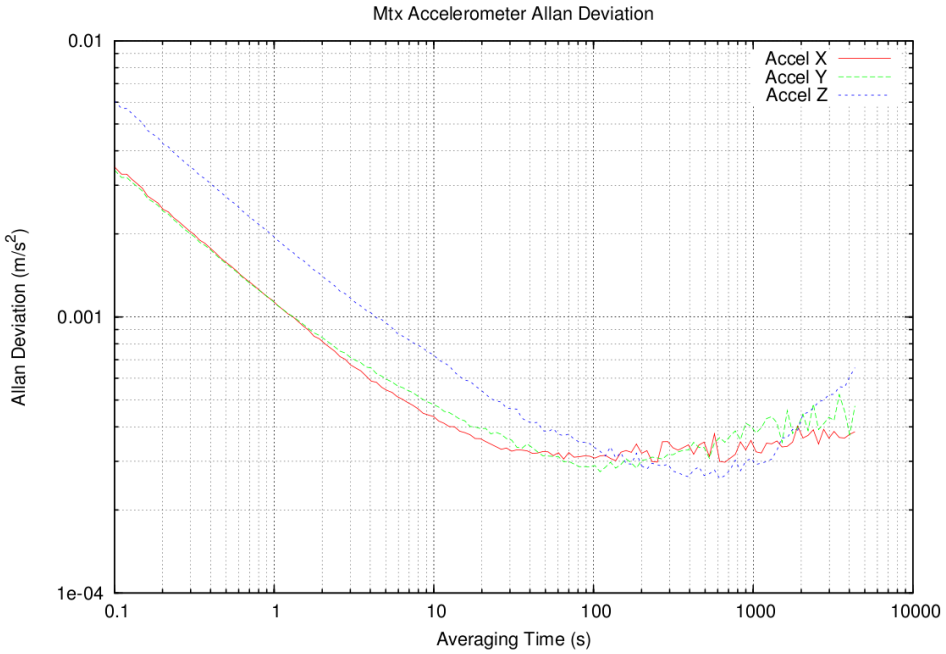

Figure 10.3: Xsens MTx Accelerometer Allan Deviation from woodman

calculated velocity by differentiating the position data from the optical tracking system and found instances of small, 'zero', velocity which we used as truth ZUPT data. Using our technique we plotted the percentage of false positives vs. detection for various window sizes in Fig. 10.4. From this data we selected the window size of ±6 samples and a threshold of .0165, chosen to keep the percentage of false positives below 1%.

10.5 Results

There are no reference points provided with which to compute an absolute initial yaw estimate so the filter's values have been rotated to minimize error in the early stages of the walk. Using our method for computing ZUPTs and the filter designed through the processes in this thesis we were able to achieve the path result seen in Fig. 10.5. This result is encouraging because it is based on a real data set of a sensor mounted to a foot, as opposed to one sitting on a desk and moved by hand. Judging from the fact that the filter achieves a track, it also uses a ZUPT detection algorithm that appears to be detecting ZUPTs reliably. It is worth noticing how the filter tracks the movements of the foot while it

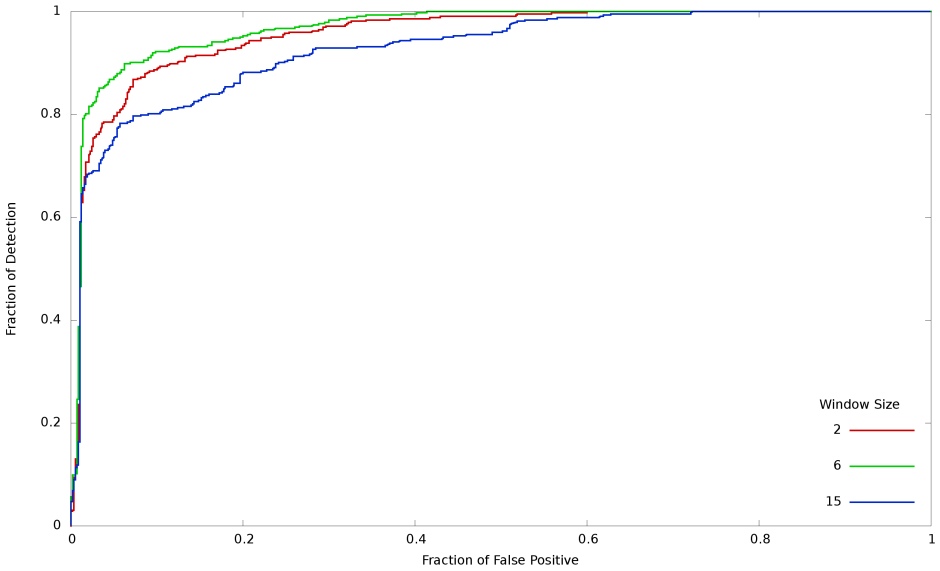

Figure 10.4: ZUPT Detection Operation Curve for Different Window Sizes

is transitioning between ZUPTs, noting that its track of the motion includes fluid rotations of the direction of motion especially in the corners.

At the end of the walk the system accumulated 0.0695 m error in the XY-plane and 0.5436 m error in the Z for a percent error per unit distance traveled of 0.23% XY, 1.84% XYZ. For our filter both the individual errors and the XY error can be seen in Fig. 10.6. In the original paper, [4], typical % XY error per unit distance traveled was approximately 1% to 2% using their own implementation of a Kalman Filter.

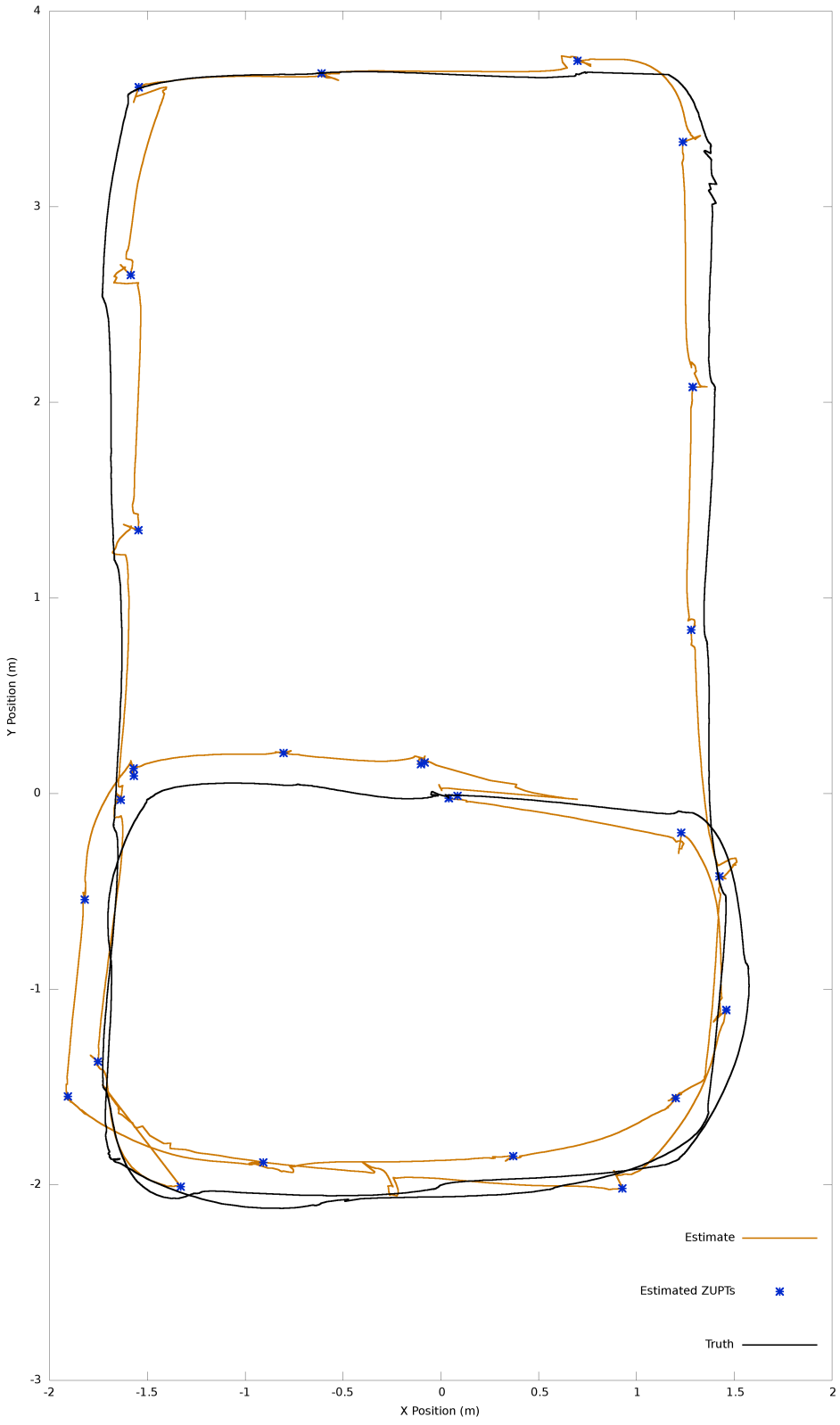

Figure 10.5: Results for the Walked Loop

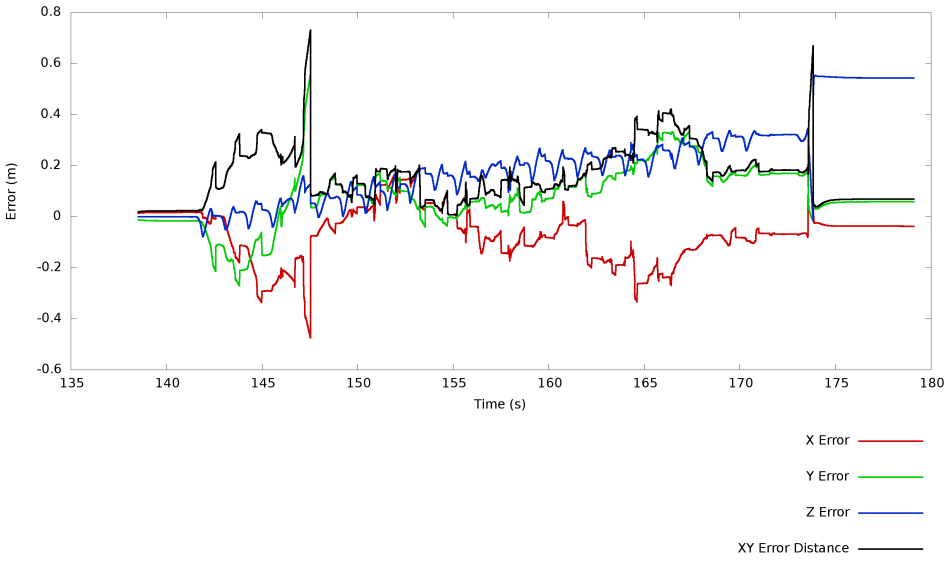

Figure 10.6: Errors for the Walked Loop

10.6 Our Pedestrian Motion Foot Mounted System Test

To further test our filter creation methodology we conducted a walking test with the Analog Devices ADIS16375 in our own lab. Instead of an infrared camera set up or surveyed ground points we simply returned to the starting point and all accumulated displacement is assumed to be error. The inertial unit was fitted with an enclosure and mounted to our RF data acquisition system and the pair were attached to the boot through a metal bracket and straps. The entire setup can be seen in Figures 10.7 and 10.8. The shoe was worn while the subject being tracked executed maneuvers through a short initialization stage where the foot was lifted and placed down a few times, and then walked in a loop around our lab. The ZUPT detection algorithm previously discussed was modified to use the small time variance of the gyroscopes but otherwise remains theoretically identical.

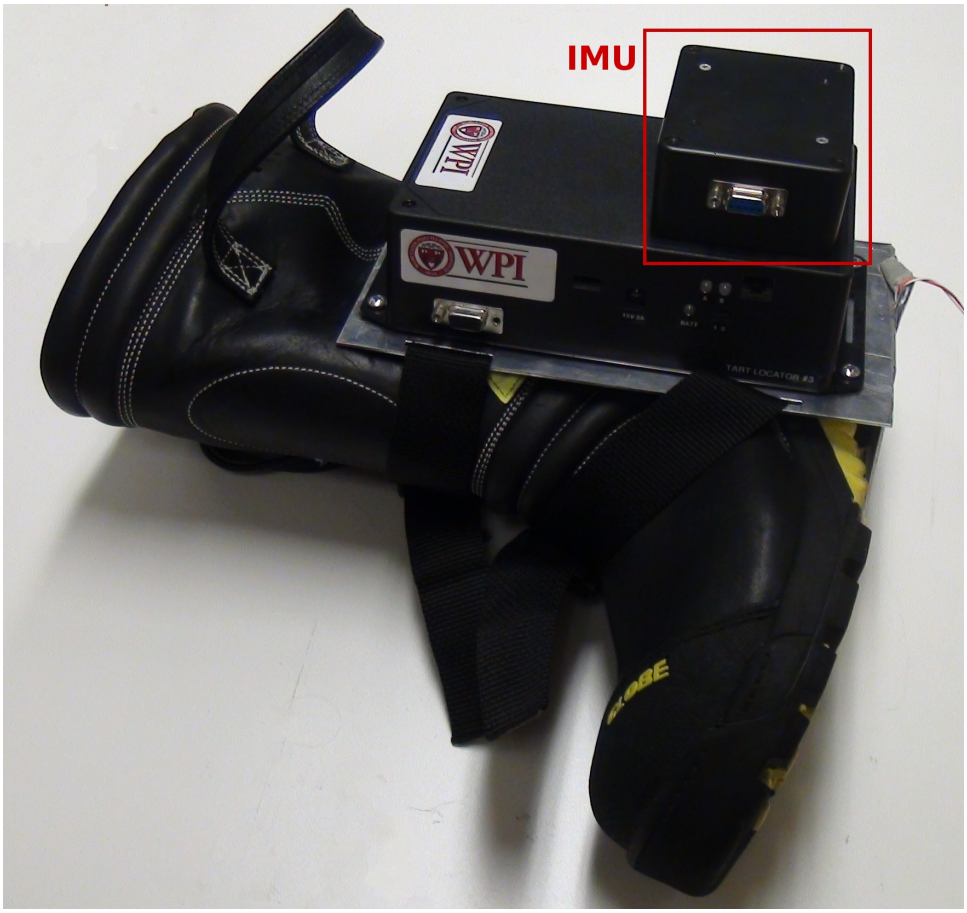

Figure 10.7: Our Sensor Boot Setup

Figure 10.8: Our Sensor Boot in Action

The resulting estimated track can be seen in Figure 10.9, the path starts at the origin in light teal and moves to dark blue. The final displacement and hence accumulated error for the track was $[-0.606, 0.014, 0.535]$ m. The entire path length is estimated at 25 m giving the test an approximate percent error of 2.42% in XY and 3.23% XYZ. For short walks of systems of this type an error of an approximately of 2% XY error per unit distance traveled a further 2% Z error per unit distance traveled is expected [13].

The large excursions one can see just before a ZUPT are due to accumulated error that are suddenly corrected right as the unit enters the ZUPT. For more details on this phenomenon see [9]. These can be made smooth with a fixed interval smoothing filter or by only displaying position during ZUPT intervals.

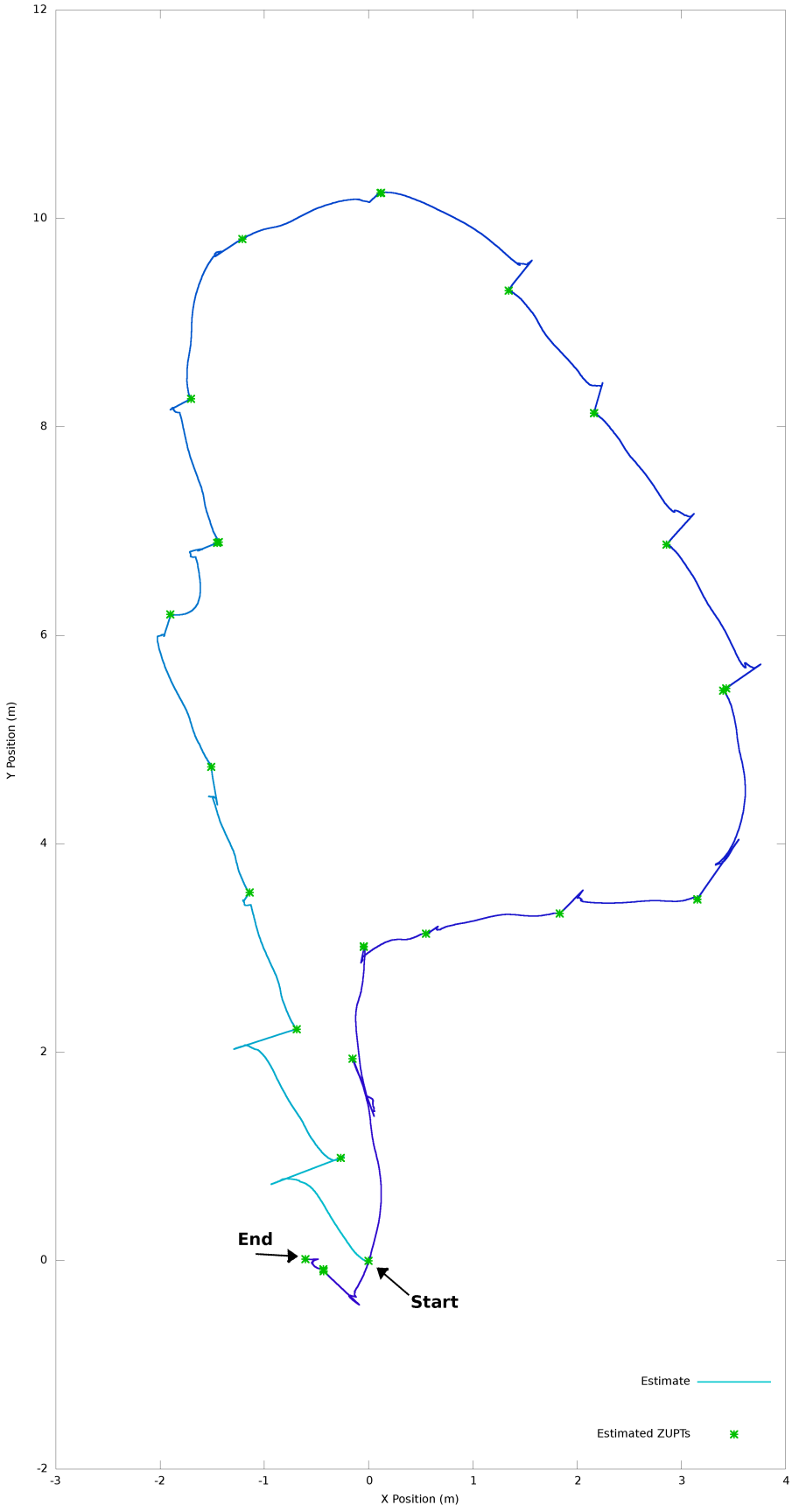

Figure 10.9: PPL Closed Loop Lab Walk with ADIS16375

Chapter 11

Conclusion

This thesis sets out to construct an inertial based tracking solution that is appropriate for potential use with technologies being developed by PPL project at WPI. In accomplishing this goal this work seeks to provide a working Kalman filter based implementation and a specification driven *method* for constructing such filters. This will be important as the project moves forward in order to integrate new and different sensors into the current implementation. We also strive to present, in one unabridged work, a set of steps for navigation and tracking systems from theory to implementation.

11.1 Contributions

In total this thesis develops a complete and self-contained method for creating Kalman filters appropriate for inertial tracking. This includes an explanation of how to construct models of the physical motion of the sensor and how to chart the rotation involved. Also examined is a method of constructing noise models for the various sensor noise effects based on values derived from specifications. The assembly of these various models is detailed, and automated through the use of Maple scripts. This is unique from the many implementations which "design" filters based on values that were tweaked in order to achieve results.

These methods were tested against actual captured data sets. The outcomes demonstrate the validity of the filter and the flexibility of the system to develop and implement models for a multitude of sensors. In the first data examined, we set captured motion data atop a table in a lab environment, where motions are kept small and simple; the filter achieved a XYZ % error per unit distance traveled of 1.63%. The second data set was a

reference data set captured by another researcher [4] in which very accurate truth data was collected through the use of an infrared video tracking setup. In this test we demonstrated a ZUPT detection system not detailed in the original reference and were able to achieve a XYZ % error per unit distance traveled of 1.84%. We then created our own instrumented boot and performed a simple walk around our own lab with a closed loop XYZ % error per unit distance traveled of 3.23%. These results achieved the state of the art in performance for similar academic and commercial systems.

This thesis creates an extendable framework which allows additional sensors to be integrated into the system with ease. We hope to be able to extend this system and use additional information from our RF system, which is under development. At its simplest this additional information would come in the form of RF position estimates, but could also include velocity or heading measurements. Additional sensors, such as barometric pressure and acoustic range, are also being explored as very natural additions to the current system.

11.2 Future Research

As was demonstrated in Appendix B, duality principles should allow the reduction of linearization errors by carrying back information from the future. This reduction of linearization errors could greatly improve the filters performance with sparse high quality information.

Not discussed in this work, but also being explored, is the potential for the filter to aid our other location system. This would be accomplished by either tracking various quantities as additional state variables that have not historically been tracked or by providing accurate short term path information to other systems. An example would be tracking the clock drift of the various RF systems to provide more robust clock alignment, or to provide accurate path estimates between RF position updates.

Not explored by this thesis in any depth but never-the-less an important and fruitful avenue of research is the construction of better models for the motion of the object being tracked. This includes, but is not limited to, the modeling of how people move as seen in [13]. Implementation of such a system may require defining additional states and better defining the innovation noise or something as simple as defining additional observations, the ZUPT being a very simple form of this modeling.

It is the hope that this work will provide a foundation for a fruitful avenue for improving

the performance of our indoor tracking systems.

Bibliography

[1] IEEE Recommended Practice for Inertial Sensor Test Equipment, Instrumentation, Data Acquisition, and Analysis. *IEEE Std 1554-2005*, 2005.

[2] Vincent Amendolare. Synchronization in an Indoor Precision Location System. Master's thesis, Worcester Polytechnic Institute, 2007.

[3] Vincent T. Amendolare. *Transactional Array Reconcilliation Tomography for Precision Indoor Location.* PhD thesis, Worcester Polytechnic Institute, 2010.

[4] Michael Angermann, Patrick Robertson, Thomas Kemptner, and Mohammed Khider. A High Precision Reference Data Set for Pedestrian Navigation using Foot-Mounted Inertial Sensors. In *International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, Zurich, Switzerland, September 2010.

[5] Matthew C. Campbell. Design of a Mobile Transceiver for Precision Indoor Location. Master's thesis, Worcester Polytechnic Institute, 2010.

[6] Andrew Cavanaugh. Bayesian information fusion for precision indoor location. Master's thesis, Worcester Polytechnic Institute, 2011.

[7] Andrew Cavanaugh, Matthew Lowe, David Cyganski, and R. James Duckworth. WPI Precision Personnel Location System: Rapid Deployment Antenna System and Supporting Algorithms for 3D Precision Location. In *ION International Technical Meeting*, January 2010.

[8] Wendell Fleming and Raymond Rishel. *Deterministic and Stochastic Optimal Control.* Springer-Verlag, New York, 1975.

[9] Eric Foxlin and Sheng Wan. Improved Pedestrian Navigation Based on Drift-Reduced MEMS IMU Chip. In *ION International Technical Meeting*, January 2010.

[10] The InterSense Website, 2011.

[11] Andrew H. Jazwinski. *Stochastic Processes and Filtering Theory.* Achademic Press, Inc., New York, 1970.

[12] Jack B. Kuipers. *Quaternions and Rotation Sequences.* Princeton University Press, Princeton, New Jersey, 1999.

[13] L. Ojeda and J. Borenstein. Personal Dead-reckoning System for GPS-denied Environments. In *Safety, Security and Rescue Robotics, 2007. SSRR 2007. IEEE International Workshop on*, pages 1 –6, September 2007.

[14] Athanasios Papoulis and S. Unnikrishna Pillai. *Probability, Random Variables and Stochastic Processes.* McGraw-Hill, New York, 2002.

[15] R. Tyrrell Rockafellar. *Convex Analysis.* Princeton University Press, Princeton, New Jersey, 1970.

[16] Robert F. Stengel. *Optimal Control and Estimation.* Dover Publications, New York, 1986.

[17] Paul Waltman. *A Second Course in Elementary Differential Equations.* Dover Publications, Mineola, New York, 2004.

[18] Oliver J. Woodman. An introduction to inertial navigation. Technical Report UCAM-CL-TR-696, University of Cambridge, Computer Laboratory, August 2007.

Appendix A

Kalman-Bucy Filter Discussion

As an effort to cast the Kalman filter in a new light, from the perspective of control theory, we may rederive its results with different principles. The effort was aided greatly by the solution to the Linear Quadratic Regulator Problem as developed in [8] and many of the notations and ideas remain the same.

A.1 Problem Description

Consider the following system for $t \in [t_0, t_1]$ which is defined in the most part from the treatment in [16].

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{F}(t)\boldsymbol{x}(t) + \boldsymbol{G}(t)\boldsymbol{u}(t) + \boldsymbol{L}(t)\boldsymbol{w}(t) \tag{A.1}$$

Here $\boldsymbol{x}(t) \in \mathbb{R}^k$ is the current state of the system. The matrix $\boldsymbol{F}(t) \in \mathbb{R}^{k\times k}$ is called the state transition matrix. The system is driven with an input $\boldsymbol{u}(t) \in \mathbb{R}^j$ and its input matrix $\boldsymbol{G}(t) \in \mathbb{R}^{k\times j}$ which are the known, and $\boldsymbol{w}(t) \in \mathbb{R}^l$, an unknown noise input while its associated matrix $\boldsymbol{L}(t) \in \mathbb{R}^{k\times l}$ is known. The observations of the system, $\boldsymbol{z}(t)$, are as described by the following equation.

$$\boldsymbol{z}(t) = \boldsymbol{H}(t)\boldsymbol{x}(t) + \boldsymbol{n}(t) \tag{A.2}$$

Here $\boldsymbol{z}(t) \in \mathbb{R}^m$ so $\boldsymbol{H}(t) \in \mathbb{R}^{m\times k}$ and $\boldsymbol{n}(t) \in \mathbb{R}^m$. The functions $\boldsymbol{u}$ and $\boldsymbol{z}$ are assumed to be known on the interval. The functions $\boldsymbol{w}$ and $\boldsymbol{n}$ are assumed to be zero mean Gaussian White Noise (GWN) processes. $\boldsymbol{w}$ is sometimes called the 'innovation' noise and it represents how the system is moved by unknowable forces. $\boldsymbol{n}$ is sometimes called 'sensor' noise and it

represents how our observation of the system is corrupted.

$$E(\boldsymbol{w}(t)) = 0 \tag{A.3a}$$

$$E\big(\boldsymbol{w}(t)\boldsymbol{w}(\tau)^{\mathrm{T}}\big) = \boldsymbol{Q}(t)\delta(t-\tau) \tag{A.3b}$$

$$E(\boldsymbol{n}(t)) = 0 \tag{A.3c}$$

$$E\big(\boldsymbol{n}(t)\boldsymbol{n}(\tau)^{\mathrm{T}}\big) = \boldsymbol{R}(t)\delta(t-\tau) \tag{A.3d}$$

The system's initial state is also an unknown Gaussian random variable.

$$E(\boldsymbol{x}(t_0)) = \hat{\boldsymbol{x}}_0 \tag{A.4a}$$

$$E\big(\boldsymbol{x}(t_0)\boldsymbol{x}(t_0)^{\mathrm{T}}\big) = \boldsymbol{P}_0 \tag{A.4b}$$

The probability density function (pdf) for any given initial condition, $\boldsymbol{x}(t_0) = \tilde{\boldsymbol{x}}_0$ is given by Equation (A.5). 'J' is chosen here to foreshadow its use in a maximization problem.

$$J(\boldsymbol{x}(t_0)) = \frac{1}{(2\pi)^{\frac{k}{2}}|\boldsymbol{P}_0|^{\frac{1}{2}}} \exp\left(-\tfrac{1}{2}\left(\boldsymbol{x}(t_0)-\hat{\boldsymbol{x}}_0\right)^{\mathrm{T}}\boldsymbol{P}_0^{-1}\left(\boldsymbol{x}(t_0)-\hat{\boldsymbol{x}}_0\right)\right) \tag{A.5}$$

It is also necessary to define a pdf for both $\boldsymbol{w}$ and $\boldsymbol{n}$. They are both Gaussian White Noise so the cost at any instant is given by the pdf of a Gaussian random variable using the $\boldsymbol{Q}(t)$ and $\boldsymbol{R}(t)$ matrices. That is to say we expect that the time 'derivative' of our final functional to be the pdf of a Gaussian random variable. We also expect the property, that given the probability on a pair of intervals $[a,b)$ and $[b,c]$ are C_1 and C_2 respectively, the probability for the total interval $[a,c]$ should be C_1C_2.

Given a noise function $\boldsymbol{y}(t)$ whose pdf at any instant, t, is given by $f(\boldsymbol{y},t)$, the following definition will meet the previous conditions on an interval $[a,c]$.

$$\mathfrak{Y}\left(\tilde{\boldsymbol{y}}\right) = \prod_a^c f(\tilde{\boldsymbol{y}},t)^{dt}$$

Here $\prod_a^b f(s)^{ds}$ is the 'product integral' defined by $\lim_{\Delta s\to 0}\prod f(s_i)^{\Delta s}$.

In our case the pdfs become as follows.

$$\mathfrak{W}(\tilde{\boldsymbol{w}}) = \prod_{t_0}^{t_1}\left[\frac{1}{(2\pi)^{\frac{l}{2}}|\boldsymbol{Q}(t)|^{\frac{1}{2}}}\exp\left(-\tfrac{1}{2}\tilde{\boldsymbol{w}}(t)^{\mathrm{T}}\boldsymbol{Q}(t)^{-1}\tilde{\boldsymbol{w}}(t)\right)\right]^{dt} \tag{A.6}$$

$$\mathfrak{N}(\tilde{\boldsymbol{n}}) = \prod_{t_0}^{t_1}\left[\frac{1}{(2\pi)^{\frac{m}{2}}|\boldsymbol{R}(t)|^{\frac{1}{2}}}\exp\left(-\tfrac{1}{2}\tilde{\boldsymbol{n}}(t)^{\mathrm{T}}\boldsymbol{R}(t)^{-1}\tilde{\boldsymbol{n}}(t)\right)\right]^{dt} \tag{A.7}$$

All the noise variables are assumed to be independent so the joint pdf for a given point $(\tilde{\boldsymbol{x}}_0, \tilde{\boldsymbol{w}}, \tilde{\boldsymbol{n}})$ is simply the product of the individual pdfs.

$$J(\tilde{\boldsymbol{x}}_0)\mathfrak{W}(\tilde{\boldsymbol{w}})\mathfrak{N}(\tilde{\boldsymbol{n}})$$

For a set of data we wish to pick a path which maximize the joint pdf. Examining the first term, J, and considering that we only wish to find the location of a maximum we may remove the leading scalar multiple.

$$J(\boldsymbol{x}(t_0)) \sim \exp\left(-\tfrac{1}{2}\left(\boldsymbol{x}(t_0) - \hat{\boldsymbol{x}}_0\right)^{\mathrm{T}} \boldsymbol{P}_0^{-1} \left(\boldsymbol{x}(t_0) - \hat{\boldsymbol{x}}_0\right)\right)$$

Expanding the second term, $\mathfrak{W}$, and using the property of the product integral, removing constant multipliers, we find.

$$\prod_a^b f(s)^{ds} = \exp\left(\int_a^b \ln\left(f(s)\right) ds\right)$$

$$\exp\left(\int_{t_0}^{t_1} \ln\left(\frac{1}{(2\pi)^{\frac{l}{2}} \left|\boldsymbol{Q}(t)\right|^{\frac{1}{2}}}\right) - \tfrac{1}{2}\tilde{\boldsymbol{w}}(t)^{\mathrm{T}}\boldsymbol{Q}(t)^{-1}\tilde{\boldsymbol{w}}(t)dt\right)$$

$$\mathfrak{W}(\tilde{\boldsymbol{w}}) \sim \exp\left(\int_{t_0}^{t_1} -\frac{1}{2}\tilde{\boldsymbol{w}}(t)^{\mathrm{T}}\boldsymbol{Q}(t)^{-1}\tilde{\boldsymbol{w}}(t)dt\right)$$

The third term, $\mathfrak{N}$, is handled in the same fashion as the second.

$$\mathfrak{N}(\tilde{\boldsymbol{n}}) \sim \exp\left(\int_{t_0}^{t_1} -\frac{1}{2}\tilde{\boldsymbol{n}}(t)^{\mathrm{T}}\boldsymbol{R}(t)^{-1}\tilde{\boldsymbol{n}}(t)dt\right)$$

Recombined the formula becomes

$$\exp\left(-\frac{1}{2}\left(\left(\boldsymbol{x}(t_0) - \hat{\boldsymbol{x}}_0\right)^{\mathrm{T}} \boldsymbol{P}_0^{-1} \left(\boldsymbol{x}(t_0) - \hat{\boldsymbol{x}}_0\right) + \int_{t_0}^{t_1} \tilde{\boldsymbol{w}}(t)^{\mathrm{T}}\boldsymbol{Q}(t)^{-1}\tilde{\boldsymbol{w}}(t) + \tilde{\boldsymbol{n}}(t)^{\mathrm{T}}\boldsymbol{R}(t)^{-1}\tilde{\boldsymbol{n}}(t)dt\right)\right)$$

A point maximizing this, the one that is most likely, will also maximize the natural log of this, because the natural log is a purely increasing.

$$-\frac{1}{2}\left(\left(\boldsymbol{x}(t_0) - \hat{\boldsymbol{x}}_0\right)^{\mathrm{T}} \boldsymbol{P}_0^{-1} \left(\boldsymbol{x}(t_0) - \hat{\boldsymbol{x}}_0\right) + \int_{t_0}^{t_1} \tilde{\boldsymbol{w}}(t)^{\mathrm{T}}\boldsymbol{Q}(t)^{-1}\tilde{\boldsymbol{w}}(t) + \tilde{\boldsymbol{n}}(t)^{\mathrm{T}}\boldsymbol{R}(t)^{-1}\tilde{\boldsymbol{n}}(t)dt\right)$$

Finally we may remove the negative sign and change the problem from maximization to minimization and define the function $\mathfrak{K}$ which we wish to minimize

$$\begin{aligned}\mathfrak{K}(\boldsymbol{x}(t_0), \tilde{\boldsymbol{w}}, \tilde{\boldsymbol{n}}) = &\left(\boldsymbol{x}(t_0) - \hat{\boldsymbol{x}}_0\right)^{\mathrm{T}} \boldsymbol{P}_0^{-1} \left(\boldsymbol{x}(t_0) - \hat{\boldsymbol{x}}_0\right) \\ &+ \int_{t_0}^{t_1} \tilde{\boldsymbol{w}}(t)^{\mathrm{T}}\boldsymbol{Q}(t)^{-1}\tilde{\boldsymbol{w}}(t) + \tilde{\boldsymbol{n}}(t)^{\mathrm{T}}\boldsymbol{R}(t)^{-1}\tilde{\boldsymbol{n}}(t)dt\end{aligned} \tag{A.8}$$

A.2 Control Problem

From the pair $(\boldsymbol{x}(t_0), \tilde{\boldsymbol{w}})$ we may solve equation (A.1) to find a solution $\tilde{\boldsymbol{x}}$ for the given $\boldsymbol{u}, \boldsymbol{G}, \boldsymbol{L}$. With this solution, $\tilde{\boldsymbol{x}}(t)$, a trivial manipulation of (A.2) results in an expression for $\tilde{\boldsymbol{n}}(t) = \boldsymbol{z}(t) - \boldsymbol{H}(t)\tilde{\boldsymbol{x}}(t)$. This simplifies the formula in (A.8) to be a minimization of a function of $(\boldsymbol{x}(t_0), \tilde{\boldsymbol{w}})$, which we will call $\mathfrak{J}$.

$$\mathfrak{J}(\boldsymbol{x}(t_0), \tilde{\boldsymbol{w}}) = \mathfrak{K}(\boldsymbol{x}(t_0), \tilde{\boldsymbol{w}}, \tilde{\boldsymbol{n}}) \tag{A.9}$$

A.2.1 The Optimal Control Problem as stated in [8]

Given a system, $\boldsymbol{r}(t) \in \mathbb{R}^q$, whose the dynamics are contained in a function $\boldsymbol{f}$,

$$\dot{\boldsymbol{r}}(t) = \boldsymbol{f}(t, \boldsymbol{r}(t), \boldsymbol{u}(t))$$

and a function $\boldsymbol{\varphi}(t)$ which contains the *performance index* and end conditions.

$$\varphi_1\big(\breve{t}_0, \breve{t}_1, \boldsymbol{r}\big(\breve{t}_0\big), \boldsymbol{r}\big(\breve{t}_1\big)\big)$$

$$\varphi_j\big(\breve{t}_0, \breve{t}_1, \boldsymbol{r}\big(\breve{t}_0\big), \boldsymbol{r}\big(\breve{t}_1\big)\big) = 0 \quad j = 2, \cdots, n$$

Where φ_1 is the performance index which we wish to minimize and while $\varphi_2, \cdots, \varphi_n$ are end conditions.

We say that $\big(\breve{t}_0, \breve{t}_1, \breve{\boldsymbol{r}}\big(\breve{t}_0\big), \breve{\boldsymbol{u}}\big)$ is the solution to the optimal control problem defined above if it minimizes φ_1 while maintaining the equalities $\varphi_2 = 0, \cdots, \varphi_n = 0$.

A.2.2 Our Control Problem

The performance index defined in (A.9) does not meet the criteria for the performance index as it is defined in terms of the function $\tilde{\boldsymbol{w}}$. The original problem we defined is a *Bolza problem* and we will need to reformulate it to a *Mayer problem*, as done in [8].
We define the first k states of $\boldsymbol{r}$ as being equivalent to the original system given in (A.1) and introduce an additional state $\boldsymbol{r}_{k+1}$ to contain dynamics of the integral term in equation (A.9). The definition for $\boldsymbol{f}$ can be found in (A.10)

$$\dot{\boldsymbol{r}}(t)_{1\cdots k} = \boldsymbol{F}(t)\boldsymbol{r}(t)_{1\cdots k} + \boldsymbol{G}(t)\boldsymbol{u}(t) + \boldsymbol{L}(t)\boldsymbol{w}(t) \tag{A.10a}$$

$$\dot{\boldsymbol{r}}(t)_{k+1} = \tilde{\boldsymbol{w}}(t)^{\mathrm{T}}\boldsymbol{Q}(t)^{-1}\tilde{\boldsymbol{w}}(t) + \tilde{\boldsymbol{n}}(t)^{\mathrm{T}}\boldsymbol{R}(t)^{-1}\tilde{\boldsymbol{n}}(t) \tag{A.10b}$$

From this we may rewrite our $\mathfrak{J}$ function as one only dependant on the start and end states of $\boldsymbol{r}(t)$ and use it as the cost function φ_1.

$$\varphi_1\big(\breve{t}_0,\breve{t}_1,\boldsymbol{r}\big(\breve{t}_0\big),\boldsymbol{r}\big(\breve{t}_1\big)\big) = \big(\boldsymbol{r}\big(\breve{t}_0\big)_{1\cdots k} - \hat{\boldsymbol{x}}_0\big)^{\mathrm{T}} \boldsymbol{P}_0^{-1} \big(\boldsymbol{r}\big(\breve{t}_0\big)_{1\cdots k} - \hat{\boldsymbol{x}}_0\big) + \boldsymbol{r}\big(\breve{t}_1\big)_{k+1} \tag{A.11}$$

The end conditions are simply that the system start at time t_0 and end at time t_1.

$$\varphi_2\big(\breve{t}_0,\breve{t}_1,\boldsymbol{r}\big(\breve{t}_0\big),\boldsymbol{r}\big(\breve{t}_1\big)\big) = \breve{t}_0 - t_0 \tag{A.12a}$$

$$\varphi_3\big(\breve{t}_0,\breve{t}_1,\boldsymbol{r}\big(\breve{t}_0\big),\boldsymbol{r}\big(\breve{t}_1\big)\big) = \breve{t}_1 - t_1 \tag{A.12b}$$

With the additional requirement that the integration term, r_{k+1} starts at zero.

$$\varphi_4\big(\breve{t}_0,\breve{t}_1,\boldsymbol{r}\big(\breve{t}_0\big),\boldsymbol{r}\big(\breve{t}_1\big)\big) = \boldsymbol{r}\big(\breve{t}_0\big)_{k+1} \tag{A.13}$$

From here we can apply the Pontryagin's Principle, a necessary condition for an optimal solution, as demonstrated in [8].

A.2.3 Pontragin's Principle as stated in [8]

There exists a non zero vector $\boldsymbol{\lambda} \in \mathbb{R}^n$ with $\lambda_1 \leq 0$ and a vector function $\boldsymbol{p}(t)$ with values in $\mathbb{R}^q$ which meet the following conditions. For simplification the point $\boldsymbol{e} = \big(\breve{t}_0,\breve{t}_1,\breve{\boldsymbol{r}}\big(\breve{t}_0\big),\breve{\boldsymbol{r}}\big(\breve{t}_1\big)\big)$ is used and g_a is taken to be the partial derivative, $g_a(x) = \frac{\partial}{\partial a} g(x)$.

$$\dot{\boldsymbol{p}}(t)^{\mathrm{T}} = \boldsymbol{p}(t)^{\mathrm{T}} \boldsymbol{f}_{\boldsymbol{r}}(t,\breve{\boldsymbol{r}}(t),\breve{\boldsymbol{u}}(t)) \tag{A.14a}$$

$$\max_{\boldsymbol{v}}\{H(t,\breve{\boldsymbol{r}}(t),\boldsymbol{v})\} = H(t,\breve{\boldsymbol{r}}(t),\breve{\boldsymbol{u}}(t)) \tag{A.14b}$$

$$\boldsymbol{p}\big(\breve{t}_1\big)^{\mathrm{T}} = \boldsymbol{\lambda}^{\mathrm{T}} \boldsymbol{\varphi}_{\boldsymbol{r}\left(\breve{t}_1\right)}(e) \tag{A.14c}$$

$$\boldsymbol{p}\big(\breve{t}_0\big)^{\mathrm{T}} = -\boldsymbol{\lambda}^{\mathrm{T}} \boldsymbol{\varphi}_{\boldsymbol{r}\left(\breve{t}_0\right)}(e) \tag{A.14d}$$

$$\boldsymbol{p}\big(\breve{t}_1\big)^{\mathrm{T}} \boldsymbol{f}\big(\breve{t}_1,\breve{\boldsymbol{r}}\big(\breve{t}_1\big),\breve{\boldsymbol{u}}\big(\breve{t}_1\big)\big) = -\boldsymbol{\lambda}^{\mathrm{T}} \boldsymbol{\varphi}_{t_1}(e) \tag{A.14e}$$

$$\boldsymbol{p}\big(\breve{t}_0\big)^{\mathrm{T}} f\big(\breve{t}_0,\breve{\boldsymbol{r}}\big(\breve{t}_0\big),\breve{\boldsymbol{u}}\big(\breve{t}_0\big)\big) = \boldsymbol{\lambda}^{\mathrm{T}} \boldsymbol{\varphi}_{t_0}(e) \tag{A.14f}$$

$$\boldsymbol{p}(t)^{\mathrm{T}} \boldsymbol{f}(t,\breve{\boldsymbol{r}}(t),\breve{\boldsymbol{u}}(t)) = \boldsymbol{\lambda}^{\mathrm{T}} \varphi_{\breve{t}_0}(e) + \int_{t_0}^{t} \boldsymbol{p}(s)^{\mathrm{T}} \boldsymbol{f}_t(s,\breve{\boldsymbol{r}}(s),\breve{\boldsymbol{u}}(s))ds \tag{A.14g}$$

Where H, the *Hamiltonian*, is defined as follows

$$H(t,\boldsymbol{a},\boldsymbol{b}) = \boldsymbol{p}(t)^{\mathrm{T}} f(t,\boldsymbol{a},\boldsymbol{b}) \tag{A.15}$$

A.2.4 Application of Pontragin's Principle

In the application of Pontragin's Principle it will be useful to observe the following identity for a symmetric matrix $\boldsymbol{K}$.

$$\frac{\partial}{\partial \boldsymbol{a}} (\boldsymbol{a}+\boldsymbol{b})^{\mathrm{T}} \boldsymbol{K} (\boldsymbol{a}+\boldsymbol{b}) = 2\boldsymbol{a}^{\mathrm{T}}\boldsymbol{K} + 2\boldsymbol{b}^{\mathrm{T}}\boldsymbol{K}$$

Plugging in our case and simplifying (A.14a). All of the following equations are simplified by the substitution of $\tilde{\boldsymbol{p}} = \boldsymbol{p}_{1\cdots k}$

$$\dot{\tilde{\boldsymbol{p}}}(t)^{\mathrm{T}} = -\tilde{\boldsymbol{p}}(t)^{\mathrm{T}}\boldsymbol{F}(t) - 2\boldsymbol{p}(t)_{k+1}\left(\boldsymbol{x}(t)^{\mathrm{T}}\boldsymbol{H}(t)^{\mathrm{T}}\boldsymbol{R}(t)^{-1}\boldsymbol{H}(t) - \boldsymbol{z}(t)^{\mathrm{T}}\boldsymbol{R}(t)^{-1}\boldsymbol{H}(t)\right)$$

$$\dot{\boldsymbol{p}}(t)_{k+1} = 0$$

Condition (A.14c) becomes

$$\tilde{\boldsymbol{p}}\breve{(t_1)}_{1\cdots k} = 0$$

$$\boldsymbol{p}\breve{(t_1)}_{k+1} = \lambda_1$$

Condition (A.14d) becomes

$$\tilde{\boldsymbol{p}}\breve{(t_0)}^{\mathrm{T}} = -2\lambda_1\left(\boldsymbol{x}(t_0)^{\mathrm{T}}\boldsymbol{P}_0^{-1} - \hat{\boldsymbol{x}}_0^{\mathrm{T}}\boldsymbol{P}_0^{-1}\right)$$

$$\boldsymbol{p}\breve{(t_0)}_{k+1} = -\lambda_4$$

Condition (A.14e) becomes

$$\boldsymbol{p}\breve{(t_1)}^{\mathrm{T}}\boldsymbol{f}\left(\breve{t_1}, \boldsymbol{r}\breve{(t_1)}, \boldsymbol{w}\breve{(t_1)}\right) = -\lambda_3$$

Condition (A.14f) becomes

$$\boldsymbol{p}\breve{(t_0)}^{\mathrm{T}}\boldsymbol{f}\left(\breve{t_0}, \boldsymbol{r}\breve{(t_0)}, \boldsymbol{w}\breve{(t_0)}\right) = \lambda_2$$

Using the same logic as page 33 of [8] we may restrict λ_1 to $\lambda_1 = -\frac{1}{2}$, forcing $\boldsymbol{p}(t)_{k+1} = -\frac{1}{2}$. Condition (A.14b) becomes

$$\begin{aligned}
&\max_{\boldsymbol{v}\in\mathbb{R}^l}\{\tilde{\boldsymbol{p}}(t)^{\mathrm{T}}\left(\boldsymbol{F}(t)\boldsymbol{x}(t) + \boldsymbol{G}(t)\boldsymbol{u}(t) + \boldsymbol{L}(t)\boldsymbol{v}\right) + \\
&\qquad \boldsymbol{p}(t)_{k+1}\left(\boldsymbol{v}^{\mathrm{T}}\boldsymbol{Q}(t)^{-1}\boldsymbol{v} + \left(\boldsymbol{z}(t) - \boldsymbol{H}(t)\boldsymbol{x}(t)\right)^{\mathrm{T}}\boldsymbol{R}(t)^{-1}\left(\boldsymbol{z}(t) - \boldsymbol{H}(t)\boldsymbol{x}(t)\right)\right)\} = \\
&\tilde{\boldsymbol{p}}(t)^{\mathrm{T}}\left(\boldsymbol{F}(t)\boldsymbol{x}(t) + \boldsymbol{G}(t)\boldsymbol{u}(t) + \boldsymbol{L}(t)\boldsymbol{w}(t)\right) + \\
&\qquad \boldsymbol{p}(t)_{k+1}\left(\boldsymbol{w}(t)^{\mathrm{T}}\boldsymbol{Q}(t)^{-1}\boldsymbol{w}(t) + \left(\boldsymbol{z}(t) - \boldsymbol{H}(t)\boldsymbol{x}(t)\right)^{\mathrm{T}}\boldsymbol{R}(t)^{-1}\left(\boldsymbol{z}(t) - \boldsymbol{H}(t)\boldsymbol{x}(t)\right)\right)
\end{aligned}$$

Which simplifies to

$$\max_{\boldsymbol{v}\in\mathbb{R}^l}\{\tilde{\boldsymbol{p}}(t)_{1\cdots k}^{\mathrm{T}}\boldsymbol{L}(t)\boldsymbol{v}+\boldsymbol{p}(t)_{k+1}\boldsymbol{v}^{\mathrm{T}}\boldsymbol{Q}(t)^{-1}\boldsymbol{v}\}= \\ \tilde{\boldsymbol{p}}(t)_{1\cdots k}^{\mathrm{T}}\boldsymbol{L}(t)\boldsymbol{w}(t)+\boldsymbol{p}(t)_{k+1}\boldsymbol{w}(t)^{\mathrm{T}}\boldsymbol{Q}(t)^{-1}\boldsymbol{w}(t)$$

We may take a derivative and set it equal to zero to find the unique solution.

$$\boldsymbol{v}=\boldsymbol{Q}(t)\boldsymbol{L}(t)^{\mathrm{T}}\tilde{\boldsymbol{p}}(t)=\boldsymbol{w}(t) \tag{A.16}$$

Plugging the optimal control from equation (A.16) into the original system description in equation (A.1) results in

$$\dot{\boldsymbol{x}}(t)=\boldsymbol{F}(t)\boldsymbol{x}(t)+\boldsymbol{G}(t)\boldsymbol{u}(t)+\boldsymbol{L}(t)\boldsymbol{Q}(t)\boldsymbol{L}(t)^{\mathrm{T}}\tilde{\boldsymbol{p}}(t) \tag{A.17}$$

With the pair of conditions

$$\tilde{\boldsymbol{p}}(t_1)=0 \tag{A.18a}$$

$$\tilde{\boldsymbol{p}}(t_0)=\boldsymbol{P}_0^{-1}\left(\boldsymbol{x}_0-\hat{\boldsymbol{x}}_0\right) \tag{A.18b}$$

The dynamics of $\tilde{\boldsymbol{p}}(t)$ can be found to be take the form of the equation

$$\begin{aligned}\dot{\tilde{\boldsymbol{p}}}(t)&=-\boldsymbol{F}(t)^{\mathrm{T}}\tilde{\boldsymbol{p}}(t)+\boldsymbol{H}(t)^{\mathrm{T}}\boldsymbol{R}(t)^{-1}\boldsymbol{H}(t)\boldsymbol{x}(t)-\boldsymbol{H}(t)^{\mathrm{T}}\boldsymbol{R}(t)^{-1}\boldsymbol{z}(t)\\&=-\left(\boldsymbol{F}(t)^{\mathrm{T}}\tilde{\boldsymbol{p}}(t)+\boldsymbol{H}(t)^{\mathrm{T}}\boldsymbol{R}(t)^{-1}\left(\boldsymbol{z}(t)-\boldsymbol{H}(t)\boldsymbol{x}(t)\right)\right)\end{aligned} \tag{A.19}$$

This becomes the composite system

$$\begin{pmatrix}\dot{\tilde{\boldsymbol{p}}}(t)\\\dot{\boldsymbol{x}}(t)\end{pmatrix}=\boldsymbol{A}(t)\begin{pmatrix}\tilde{\boldsymbol{p}}(t)\\\boldsymbol{x}(t)\end{pmatrix}+\boldsymbol{B}(t)\begin{pmatrix}\boldsymbol{z}(t)\\\boldsymbol{u}(t)\end{pmatrix} \tag{A.20}$$

With

$$\boldsymbol{A}(t)=\begin{pmatrix}-\boldsymbol{F}(t)^{\mathrm{T}}&\boldsymbol{H}(t)^{\mathrm{T}}\boldsymbol{R}(t)^{-1}\boldsymbol{H}(t)\\\boldsymbol{L}(t)\boldsymbol{Q}(t)\boldsymbol{L}(t)^{\mathrm{T}}&\boldsymbol{F}(t)\end{pmatrix} \tag{A.21a}$$

$$\boldsymbol{B}(t)=\begin{pmatrix}-\boldsymbol{H}(t)^{\mathrm{T}}\boldsymbol{R}(t)^{-1}&\boldsymbol{0}\\\boldsymbol{0}&\boldsymbol{G}(t)\end{pmatrix} \tag{A.21b}$$

From [17] we know if $\boldsymbol{\Phi}(t)$ is a fundamental matrix then the solution to Equation (A.20) can be written as follows.

$$\begin{pmatrix}\tilde{\boldsymbol{p}}(t)\\\boldsymbol{x}(t)\end{pmatrix}=\boldsymbol{\Phi}(t)\boldsymbol{\Phi}(t_0)^{-1}\begin{pmatrix}\tilde{\boldsymbol{p}}_0\\\boldsymbol{x}_0\end{pmatrix}+\boldsymbol{\Phi}(t)\int_{t_0}^{t}\boldsymbol{\Phi}(s)^{-1}\boldsymbol{B}(s)\begin{pmatrix}\boldsymbol{z}(s)\\\boldsymbol{u}(s)\end{pmatrix}ds$$

Given the final result of an optimal solution $\boldsymbol{x}(t_1)$ it will be useful to simply correct it as future data comes in. In order to preserve the optimality we require that $\tilde{\boldsymbol{p}}(t)$ remains 0, that $\dot{\tilde{\boldsymbol{p}}}(t) = 0$. From (A.20) we can see that the derivative as it stands will not meet this criterion, so we consider varying/correcting the initial condition $\boldsymbol{x}(t_0)$ with time.
The dynamics becomes

$$\begin{pmatrix} \dot{\tilde{\boldsymbol{p}}}(t) \\ \dot{\boldsymbol{x}}(t) \end{pmatrix} = \boldsymbol{A}(t) \begin{pmatrix} \tilde{\boldsymbol{p}}(t) \\ \boldsymbol{x}(t) \end{pmatrix} + \boldsymbol{B}(t) \begin{pmatrix} \boldsymbol{z}(t) \\ \boldsymbol{u}(t) \end{pmatrix} + \boldsymbol{\Phi}(t)\boldsymbol{\Phi}(t_0)^{-1} \begin{pmatrix} \boldsymbol{P}_0^{-1} \\ \boldsymbol{I} \end{pmatrix} \dot{\boldsymbol{x}(t_0)}(t)$$

If we are at an optimal solution $\tilde{\boldsymbol{p}}(t) = 0$ and this simplifies to

$$\begin{pmatrix} \dot{\tilde{\boldsymbol{p}}}(t) \\ \dot{\boldsymbol{x}}(t) \end{pmatrix} = \begin{pmatrix} -\boldsymbol{H}(t)^{\mathrm{T}}\boldsymbol{R}(t)^{-1}\left(\boldsymbol{z}(t) - \boldsymbol{H}(t)\boldsymbol{x}(t)\right) \\ \boldsymbol{F}(t)\boldsymbol{x}(t) + \boldsymbol{G}(t)\boldsymbol{u}(t) \end{pmatrix} + \boldsymbol{\Phi}(t)\boldsymbol{\Phi}(t_0)^{-1} \begin{pmatrix} \boldsymbol{P}_0^{-1} \\ \boldsymbol{I} \end{pmatrix} \dot{\boldsymbol{x}(t_0)}(t)$$

Writing $\boldsymbol{\Phi}(t)\boldsymbol{\Phi}(t_0)^{-1}$ in a block form

$$\boldsymbol{\Phi}(t)\boldsymbol{\Phi}(t_0)^{-1} = \begin{pmatrix} \boldsymbol{\mathcal{A}}(t) & \boldsymbol{\mathcal{B}}(t) \\ \boldsymbol{\mathcal{C}}(t) & \boldsymbol{\mathcal{D}}(t) \end{pmatrix}$$

Our system becomes

$$\begin{pmatrix} \dot{\tilde{\boldsymbol{p}}}(t) \\ \dot{\boldsymbol{x}}(t) \end{pmatrix} = \begin{pmatrix} -\boldsymbol{H}(t)^{\mathrm{T}}\boldsymbol{R}(t)^{-1}\left(\boldsymbol{z}(t) - \boldsymbol{H}(t)\boldsymbol{x}(t)\right) \\ \boldsymbol{F}(t)\boldsymbol{x}(t) + \boldsymbol{G}(t)\boldsymbol{u}(t) \end{pmatrix} + \begin{pmatrix} \boldsymbol{\mathcal{A}}(t)\boldsymbol{P}_0^{-1} + \boldsymbol{\mathcal{B}}(t) \\ \boldsymbol{\mathcal{C}}(t)\boldsymbol{P}_0^{-1} + \boldsymbol{\mathcal{D}}(t) \end{pmatrix} \dot{\boldsymbol{x}(t_0)}(t)$$

The requirement $\dot{\tilde{\boldsymbol{p}}}(t) = 0$ can be written as,

$$\boldsymbol{H}(t)^{\mathrm{T}}\boldsymbol{R}(t)^{-1}\left(\boldsymbol{z}(t) - \boldsymbol{H}(t)\boldsymbol{x}(t)\right) = \left(\boldsymbol{\mathcal{A}}(t)\boldsymbol{P}_0^{-1} + \boldsymbol{\mathcal{B}}(t)\right)\dot{\boldsymbol{x}}_0(t)$$

So,

$$\dot{\boldsymbol{x}(t_0)}(t) = \left(\boldsymbol{\mathcal{A}}(t)\boldsymbol{P}_0^{-1} + \boldsymbol{\mathcal{B}}(t)\right)^{-1}\boldsymbol{H}(t)^{\mathrm{T}}\boldsymbol{R}(t)^{-1}\left(\boldsymbol{z}(t) - \boldsymbol{H}(t)\boldsymbol{x}(t)\right)$$

If we restrict ourselves to only the current value of $\boldsymbol{x}(t)$ we can plug this back in

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{F}(t)\boldsymbol{x}(t) + \boldsymbol{G}(t)\boldsymbol{u}(t) + \boldsymbol{K}(t)\left(\boldsymbol{z}(t) - \boldsymbol{H}(t)\boldsymbol{x}(t)\right) \tag{A.22}$$

where

$$\begin{aligned} \boldsymbol{P}(t) &= \left(\boldsymbol{\mathcal{C}}(t)\boldsymbol{P}_0^{-1} + \boldsymbol{\mathcal{D}}(t)\right)\left(\boldsymbol{\mathcal{A}}(t)\boldsymbol{P}_0^{-1} + \boldsymbol{\mathcal{B}}(t)\right)^{-1} \\ \boldsymbol{K}(t) &= \boldsymbol{P}(t)\boldsymbol{H}(t)^{\mathrm{T}}\boldsymbol{R}(t)^{-1} \end{aligned} \tag{A.23}$$

We'll need to track the quantity $\boldsymbol{P}(t)$ so we set up a differential equation for this purpose, recalling identities for the derivative of the matrix inverse,

$$\begin{aligned}\dot{\boldsymbol{P}}_{\boldsymbol{e}}(t) = &\left(\dot{\boldsymbol{\mathcal{C}}}(t)\boldsymbol{P}_0^{-1} + \dot{\boldsymbol{\mathcal{D}}}(t)\right)\left(\boldsymbol{\mathcal{A}}(t)\boldsymbol{P}_0^{-1} + \boldsymbol{\mathcal{B}}(t)\right)^{-1} \\ &- \left(\boldsymbol{\mathcal{C}}(t)\boldsymbol{P}_0^{-1} + \boldsymbol{\mathcal{D}}(t)\right)\left(\boldsymbol{\mathcal{A}}(t)\boldsymbol{P}_0^{-1} + \boldsymbol{\mathcal{B}}(t)\right)^{-1}\left(\dot{\boldsymbol{\mathcal{A}}}(t)\boldsymbol{P}_0^{-1} + \dot{\boldsymbol{\mathcal{B}}}(t)\right)\left(\boldsymbol{\mathcal{A}}(t)\boldsymbol{P}_0^{-1} + \boldsymbol{\mathcal{B}}(t)\right)^{-1}\end{aligned}$$

Using the condition on $\boldsymbol{\Phi}(t)$ from the original differential equation we can solve for the derivative terms

$$\dot{\boldsymbol{\Phi}}(t) = \begin{pmatrix}\dot{\boldsymbol{\mathcal{A}}}(t) & \dot{\boldsymbol{\mathcal{B}}}(t) \\ \dot{\boldsymbol{\mathcal{C}}}(t) & \dot{\boldsymbol{\mathcal{D}}}(t)\end{pmatrix} = \begin{pmatrix}-\boldsymbol{F}(t)^{\mathrm{T}} & \boldsymbol{H}(t)^{\mathrm{T}}\boldsymbol{R}(t)^{-1}\boldsymbol{H}(t) \\ \boldsymbol{L}(t)\boldsymbol{Q}(t)\boldsymbol{L}(t)^{\mathrm{T}} & \boldsymbol{F}(t)\end{pmatrix}\begin{pmatrix}\boldsymbol{\mathcal{A}}(t) & \boldsymbol{\mathcal{B}}(t) \\ \boldsymbol{\mathcal{C}}(t) & \boldsymbol{\mathcal{D}}(t)\end{pmatrix}$$

Plugging these identities in we get

$$\begin{aligned}\dot{\boldsymbol{P}}(t) = &\boldsymbol{F}(t)\boldsymbol{P}(t) + \boldsymbol{P}(t)\boldsymbol{F}(t)^{\mathrm{T}} + \boldsymbol{L}(t)\boldsymbol{Q}(t)\boldsymbol{L}(t)^{\mathrm{T}} \\ &- \boldsymbol{P}(t)\boldsymbol{H}(t)^{\mathrm{T}}\boldsymbol{R}(t)^{-1}\boldsymbol{H}(t)\boldsymbol{P}(t)\end{aligned} \tag{A.24}$$

By the identity $\boldsymbol{\Phi}(t_0)\boldsymbol{\Phi}(t_0) = \boldsymbol{I}$ we find the following initial condition

$$\boldsymbol{P}(t_0) = \boldsymbol{P}_0 \tag{A.25}$$

This result is the Kalman-Bucy filter.

A.2.5 Duality

From the fact that Pontragin's Principle based duality we can find the dual problem, the Linear Quadratic Regulator, in the vector $\tilde{\boldsymbol{p}}(t_1 - \tau)$. Page 236-237 of [16] has additional discussion of this relationship.

A.3 Demonstration

It is worth noting that there is a significant difference between the estimates of the filter during the interval and final optimal path that would have resulted in the final solution. In order to demonstrate this consider the following scenario.

A feather is falling at its terminal velocity, $v = 1$, and at time $t = 0$ the feather is at height $h(0) = 1$. Additionally the fall velocity is disturbed by wind noise.

$$\dot{p}(t) = 1 - v + w(t)$$

Where $w(t)$ is a GWN with variance 1. For the demonstration we will be estimating the noise processes as Fourier series with $M = 150$

$$a_0 + \sum_{i=1}^{M} \left(a_i \cos\left(2\pi i t\right) + b_i \sin\left(2\pi i t\right)\right)$$

At time $t = 0$ we make a measurement the feather's height with variance 1 as $\hat{h}(0) = .686$ and its velocity with variance $1/4$ as $\hat{v} = .4505000000$.

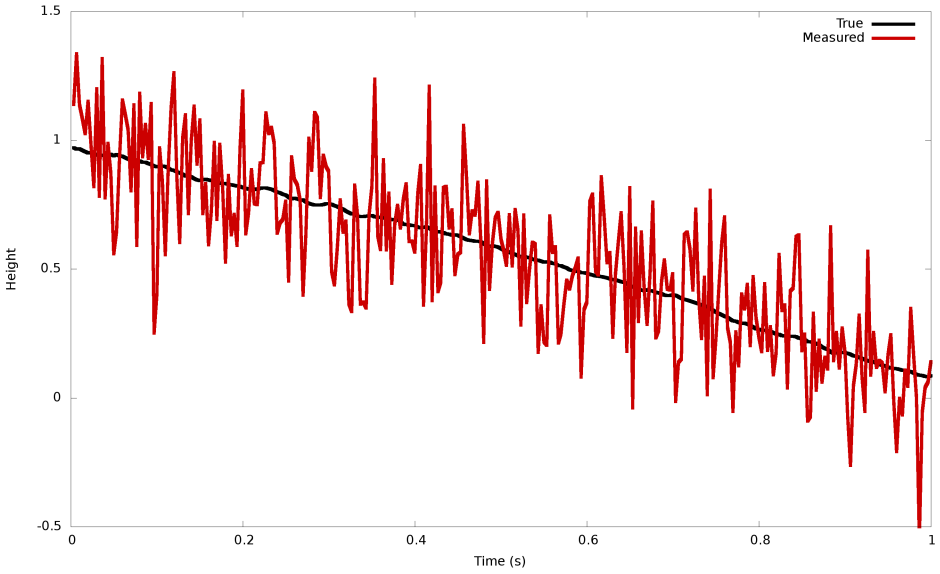

Figure A.1: Measured Feather Height

We then point a sensor at the feather which can measure its height with a Gaussian error of variance $1/25$. The measurement and truth height can be seen in Figure A.1.

This information is fed into a Kalman-Bucy filter and at the end of the interval we can construct the optimal solution, both the Kalman-Bucy's estimates and the optimal track can be seen in Figure A.2.

A.4 Advantages

It should become immediately obvious that this form of the filter gives us a new meaning for $\boldsymbol{P}(t)$ which is different from its roll as a covariance matrix in the typical filter, here it represents how the dual problems are connected as opposed to the uncertainty of the current

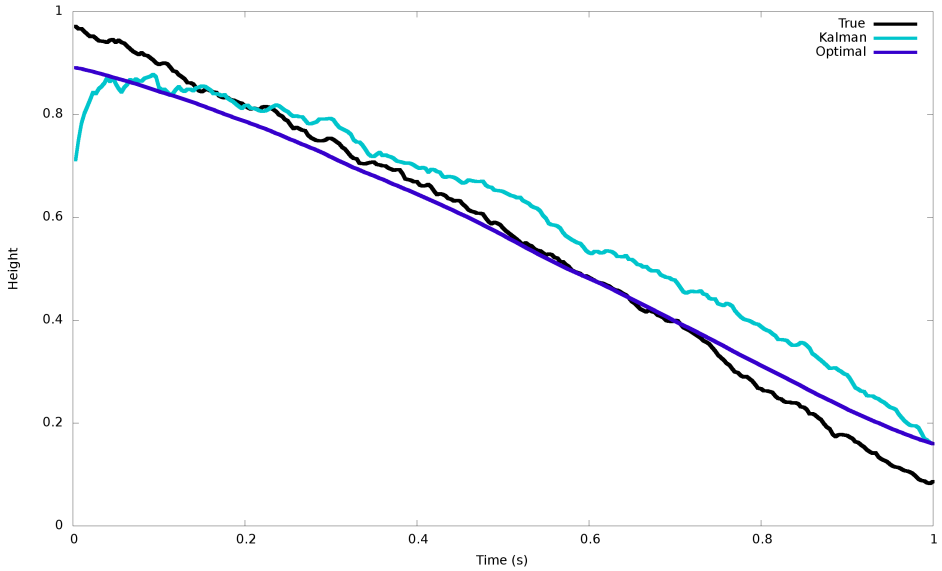

Figure A.2: Estimated Feather Height

estimate. Additionally this new filter has the ability to track non-optimal corrections in such a way as to allow them to be corrected later. This could have a host of applications, including running the filter with infrequent correction steps without any long term effects. Also, many of the advantages found in the similar realization of discrete version of the filter shown in Chapter 6 can be redeveloped for use in the continuous time filter.

Appendix B

Kalman Filtering of Nonlinear Systems

As we have seen in Chapter 6 it is possible to run the Kalman filter while making corrections to an initial state and then re-run the data with the corrected initial state in order to get a rectroactively optimal solution. In the case of a linearized system filter when we re-run the filter the result should be closer to the truth data and we should be able to pick better linearization points. With this in mind we can demonstrate the majority of this process with a linear system to illustrate this process.

B.1 Linear Filter Example

Consider the following system (similar to the one demonstrated in Appendix A)

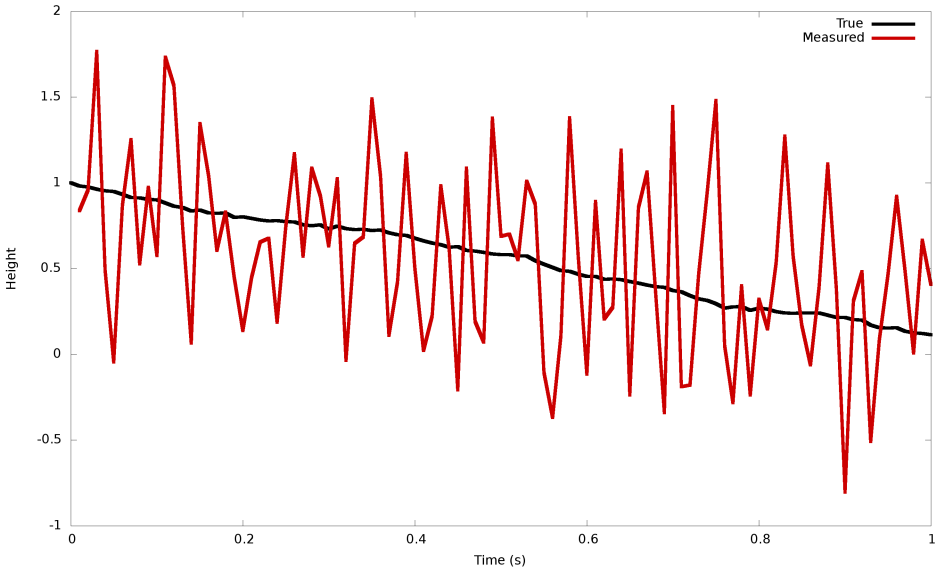

Figure B.1: Linear System's Measured Height

$$\boldsymbol{\Phi} = \begin{pmatrix} 1 & -\frac{1}{100} \\ 0 & 1 \end{pmatrix}$$

$$\boldsymbol{\Lambda} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

$$\boldsymbol{\Gamma} = 0$$

$$\boldsymbol{H} = \begin{pmatrix} 1 & 0 \end{pmatrix}$$

$$\boldsymbol{Q} = \begin{pmatrix} 0.0001 \end{pmatrix}$$

$$\boldsymbol{R} = \begin{pmatrix} 0.2 \end{pmatrix}$$

$$\boldsymbol{P}_0 = \begin{pmatrix} 1 & 0 \\ 0 & 0.125 \end{pmatrix}$$

$$\boldsymbol{x}[0] = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

A simulation of its behaviour is shown in Figure B.1 with noise drawn from appropriate Gaussian distributions

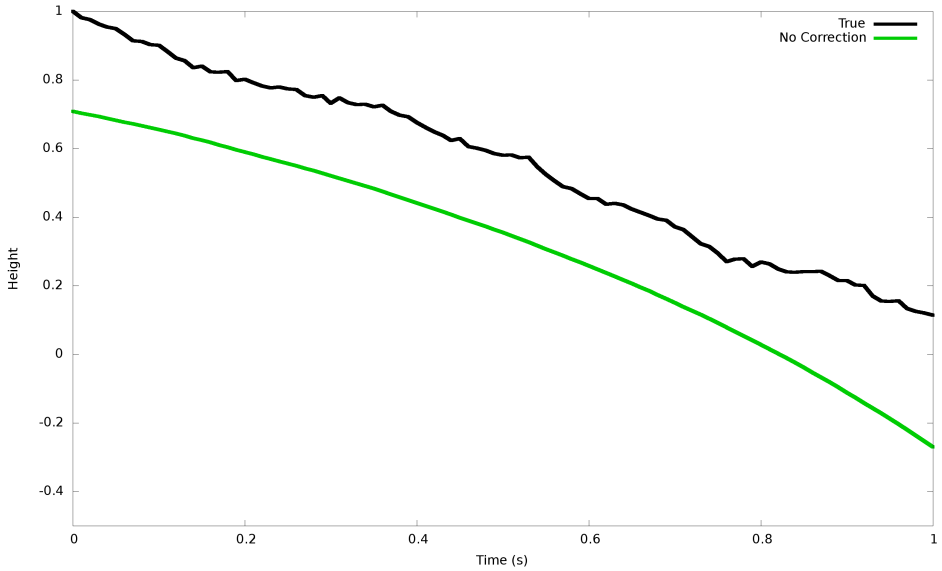

Figure B.2: Linear System's uncorrected Height Estimate

Initializing the filter at the measured initial state of $\tilde{\boldsymbol{x}}[0] = (0.708, 0.48558)$ and letting it run with measurements but without performing the correction from Eq. (6.11) we can see the resulting track of $\boldsymbol{x}_1$ in Figure B.2. This can be taken as an example of what would happen if the filter was given a non-optimal initial condition and simply left to run. The system simply accumulates errors to be corrected at the end. The accumulating of errors in the values for $\boldsymbol{\lambda}$ shown in Figure B.3, recalling that the process noise was estimated as $\tilde{\boldsymbol{w}}[i] = \boldsymbol{Q}[i]\boldsymbol{\Lambda}[i]\tilde{\boldsymbol{\lambda}}[i]$ and we expect this to be WGN.

This behaviour of simply accumulation the errors and correcting for them at the end is not desirable for a linearized system as the errors compound in ways the system cannot correct.

Re-running the data after using Eq. (6.11) to update the initial state of the filter for the accumulated error results in an optimal solution. From this run-through of the data we can learn what to expect when using the same linearization points as the run where the current optimal initial condition was calculated.

Notice we have a much better estimate for what $\boldsymbol{\lambda}$ should look like as seen in Figure B.4. So when we re-run the filter picking new linearization points we can correct $\boldsymbol{\lambda}$ to these values instead of letting it accumulate as in Figure B.3 while picking new linearization points. Performing this on a linearized system results in the optimal solution again, as the

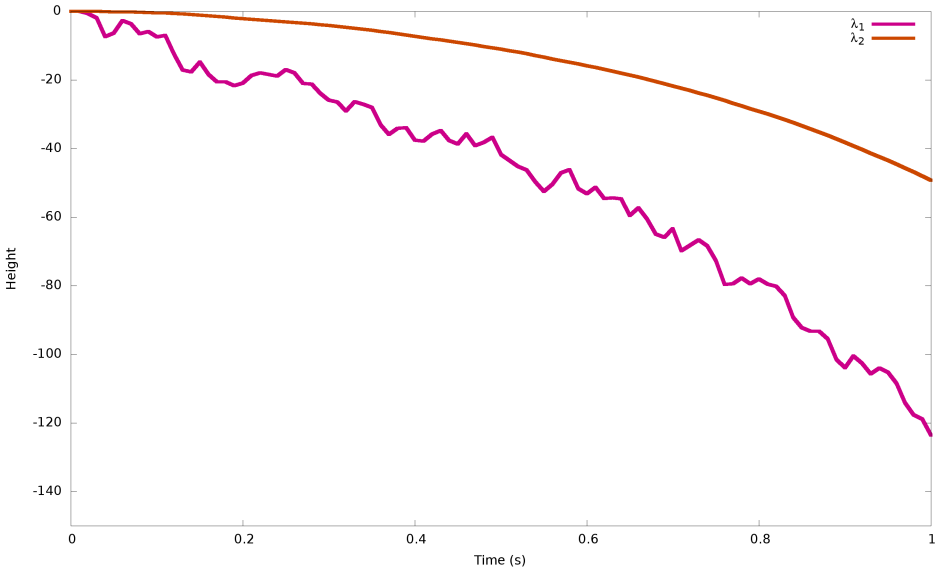

Figure B.3: Linear System's uncorrected Dual Estimate

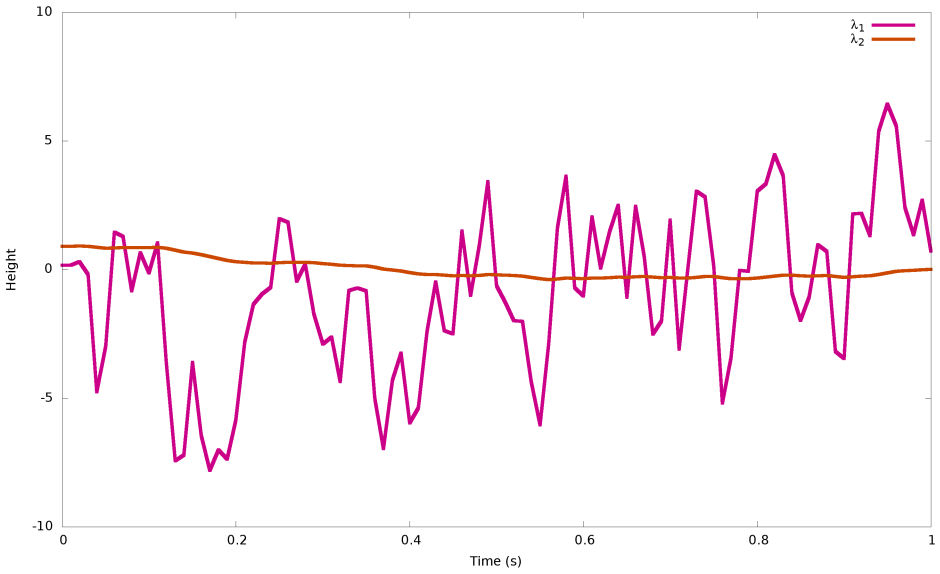

Figure B.4: Linear System's Optimal Dual Estimate

linear system is not dependant on linearization points, *but on a nonlinear system can result in a better solution as we will see.*

B.2 Nonlinear Filter Example

Consider the nonlinear system defined below.

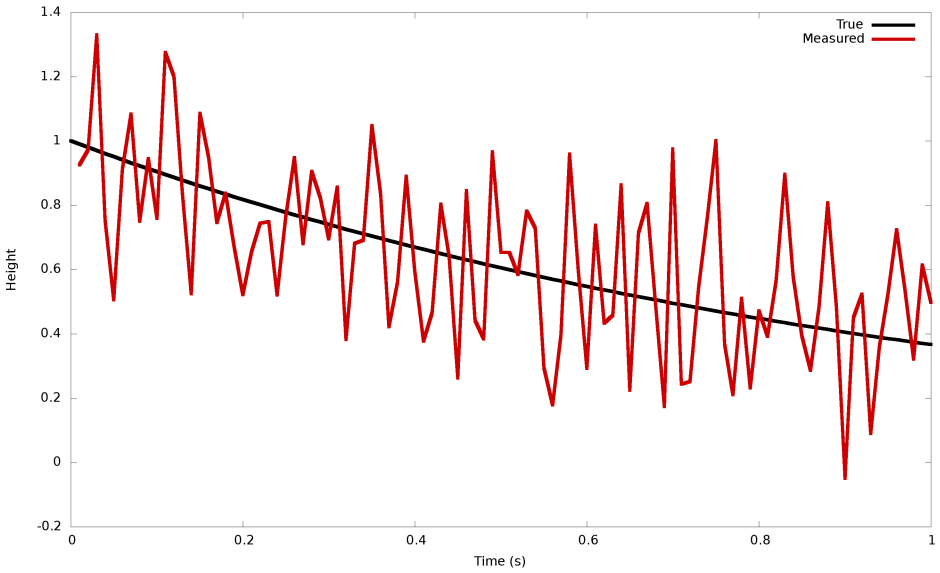

Figure B.5: Nonlinear System's Measured Height

$$
\begin{aligned}
\boldsymbol{x}[i+1]_1 =& \boldsymbol{x}[i]_1 - \frac{1}{100}\boldsymbol{x}[i]_1\boldsymbol{x}[i]_2\boldsymbol{x}[i]_3\boldsymbol{x}[i]_3 \\
\boldsymbol{x}[i+1]_2 =& \boldsymbol{x}[i]_2 \\
\boldsymbol{x}[i+1]_3 =& \boldsymbol{x}[i]_3
\end{aligned}
$$

$$
\boldsymbol{\Lambda} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}
$$

$$
\boldsymbol{\Gamma} = 0
$$

$$
\boldsymbol{H} = \begin{pmatrix} 1 & 0 & 0 \end{pmatrix}
$$

$$
\boldsymbol{Q} = \begin{pmatrix} 1e-08 \end{pmatrix}
$$

$$
\boldsymbol{R} = \begin{pmatrix} 0.04 \end{pmatrix}
$$

$$
\boldsymbol{P}_0 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0.125 & 0 \\ 0 & 0 & 0.125 \end{pmatrix}
$$

$$
\boldsymbol{x}[0] = \begin{pmatrix} 1 \\ 2 \\ 0.70711 \end{pmatrix}
$$

With the additional observation at the end, $i = 100$, to demonstrate how information gained at the end can impact how we estimate the other states through the relinearization process.

$$
\boldsymbol{H} = \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}
$$

$$
\boldsymbol{R} = \begin{pmatrix} 0.0001 \end{pmatrix}
$$

The initial state is measured as $(0.708, 1.4856, 0.1941)$, the additional observation at the end is 0.72203, and the observed height is shown in Figure B.5.

The output of a standard Kalman filter and its companion optimal solution can be seen in Figure B.6.

Running the system again relinearizing and correcting to the $\boldsymbol{\lambda}$ generated in the previous optimal solution we generate a new result also shown in Figure B.6. With this we can

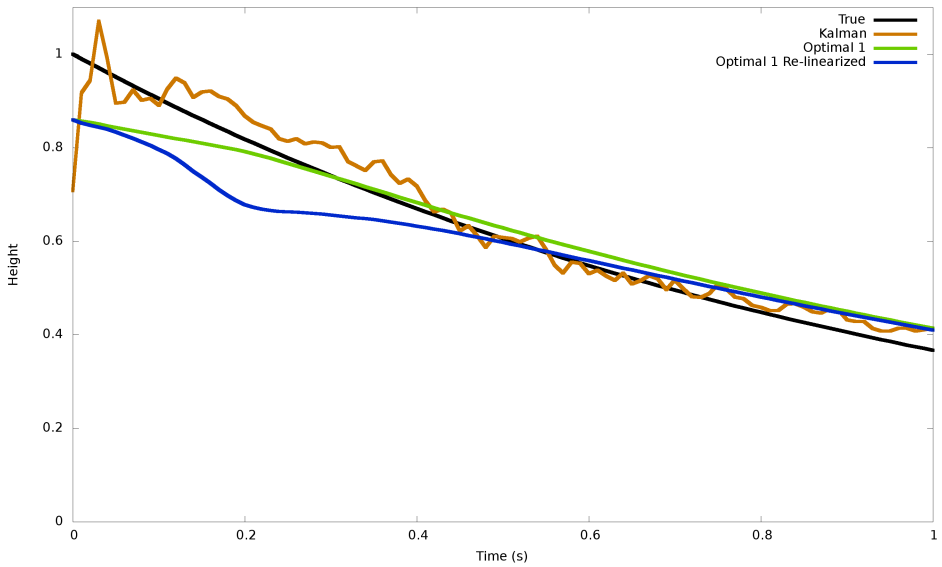

Figure B.6: Nonlinear System's First Pass Estimated Height

generate a new linear optimal solution/relinearization pair which can be seen in Figure B.7. Continuing this process on this particular system yields very marginal gains.

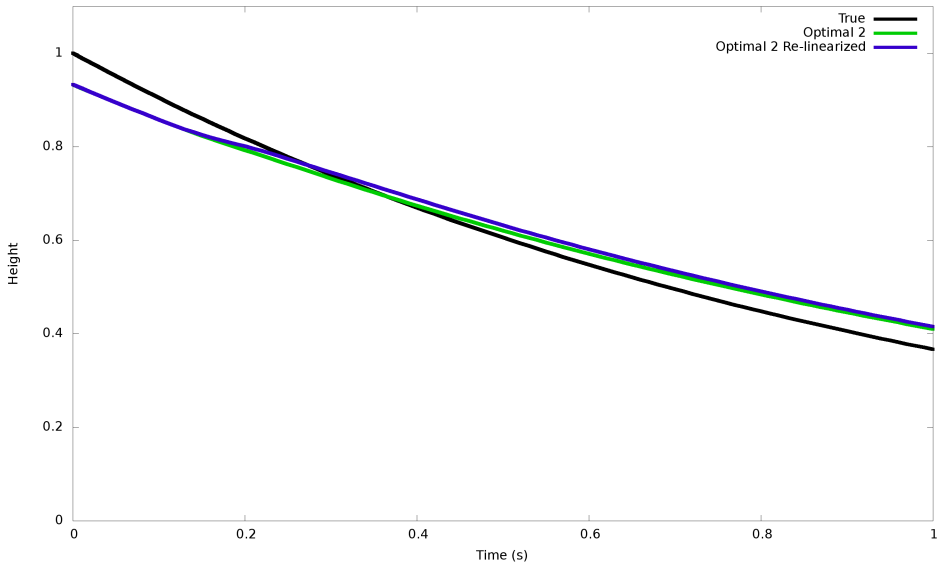

Figure B.7: Nonlinear System's Second Pass Estimated Height